MASTER EN SCIENCES INFORMATIQUES

Gestion électronique de documents multimédia: techniques de recherche et d'indexation

Anciaux, Julie

Award date:
2004

Facultés Universitaires Notre-Dame de la Paix
Institut d'Informatique

Gestion Electronique de Documents Multimédia: Techniques de Recherche et d'Indexation

Julie Anciaux

Mémoire présenté en vue de l'obtention du grade de Maître en Informatique.

Année Académique 2003-2004

RÉSUMÉ

De nos jours, la quantité de documents électroniques, de toute nature et de tout format, ne cesse de croître. En particulier, les documents multimédia se font de plus en plus nombreux et requièrent donc une gestion efficace. En effet, dans bien des domaines, il est essentiel de disposer d'un outil capable de récupérer facilement des documents multimédias sur base de critères particuliers. Les moteurs de recherche courants, tels que ceux que l'on peut voir sur Internet, permettent uniquement de récupérer des documents sur base de critères externes au document (tels que l'auteur, l'extension, le titre ou la date de création). Or, dans le domaine de l'audiovisuel, il serait préférable de pouvoir rechercher les documents sur base de leur description (i.e. ce que l'on voit sur la photo ou ce que l'on entend dans l'enregistrement sonore). Ce mémoire se propose d'aborder le cas de la gestion électronique des documents multimédia, en se concentrant sur les techniques permettant de les indexer et de les récupérer. Il explore deux approches d'indexation et de recherche, pouvant toutes les deux se baser sur les descriptions MPEG-7: la première utilise les mots-clés de la description du document multimédia et la seconde utilise son contenu.

Mots-clés: documents multimédia, moteur de recherche, indexation, recherche par mots-clés, recherche par le contenu, MPEG-7.

ABSTRACT

Nowadays the amount of electronic documents, of all kinds and every format, does not stop growing. Particularly, the multimedia documents are becoming more and more numerous and call for an effective management. Indeed, in many fields it is essential to have a tool able to retrieve easily multimedia documents on the basis of particular criteria. The common query engines, such as the ones we can see on the Internet, only allow to retrieve documents on the basis of criteria external to this document (like the author, extent, title or creation date). Yet, in the case of the audiovisual field, it would be better to be able to retrieve the documents on their description basis (i.e. what we can see on the picture, or what we can hear on the sound recording). This thesis intends to tackle the case of the electronic management of multimedia documents, focusing on the techniques which allow to index and to retrieve them. It examines two approaches of indexing and retrieval, both can be based on the MPEG-7 descriptions: the first one uses the text-based retrieval of the multimedia document's description and the second one its content-based retrieval.

Keywords: multimedia document, query engine, indexing, text-based retrieval, content-based retrieval, MPEG-7.

Je tiens à remercier toutes les personnes sans qui ce mémoire n'aurait pas pu être réalisé…

Tout d'abord, le Professeur Jean-Marie Jacquet, pour ses précieux conseils et sa disponibilité.
Toute l'équipe d'Arafox, et particulièrement Pascal Maurieras, pour m'avoir accueillie dans leur société lors de mon stage de fin d'études.
Mes parents pour m'avoir permis d'en arriver là, malgré les Sœurs Clarisse, et mes deux frères pour leurs divertissements lors de la rédaction de mon mémoire.
Ma meilleure amie, Chris, pour avoir effectué ce petit boulot de traduction.
Et, bien sûr, Anthony pour son soutien pendant toutes ces années d'études, ses encouragements, et pour tout le reste.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

Introduction

De nos jours, la quantité de documents électroniques est devenue impressionnante. En particulier, les documents audiovisuels prennent une place de plus en plus importante dans notre vie. Pensons notamment aux nombreuses pages Web parcourues chaque jour, comprenant principalement de l'information textuelle, mais également des images, du son, et des vidéos. Un autre exemple est la quantité d'archives audiovisuelles récoltées par les musées ou les chaînes de télévision telles que la RTBF, qui sont appelées à être récupérées facilement, rapidement, et en fonction de critères particuliers. Il se pourrait, par exemple que l'on désire retrouver toutes les archives vidéo dans lesquelles on apercevrait Jacques Brel. Afin d'éviter à l'utilisateur de parcourir toutes les vidéos, une à une, pour voir si elle contiennent le chanteur en question, il serait intéressant (voire crucial) de disposer d'un système permettant de récupérer les documents à l'aide d'une simple requête. Celle-ci pourrait être, par exemple, "retrouvez, s'il vous plait, toutes les vidéos dans lesquelles on peut apercevoir Jacques Brel". Evidemment, pour pouvoir récupérer facilement les documents audiovisuels, il est impératif que ceux-ci aient été indexés de manière efficace.

Ce mémoire se propose d'aborder le cas de la gestion électronique des documents audiovisuels et, plus particulièrement, la manière dont ceux-ci peuvent être indexés et récupérés.

Qui dit "gestion électronique de documents multimédia", dit "gestion électronique de documents", avec tous ses avantages (gain de place, facilité d'accès) et ses inconvénients (les problèmes d'authenticité et d'intégrité du document), que nous aborderons dans le premier chapitre de ce mémoire. Nous nous intéresserons également à l'organisation d'un système de gestion électronique de documents et à ses différentes fonctions (acquisition, restitution et diffusion de documents).

La particularité de la gestion électronique de documents multimédia vient précisément du caractère audiovisuel des documents en question. Comme nous le verrons dans le deuxième chapitre de ce mémoire, l'information multimédia se compose de textes, d'images (fixes ou animées) et/ou de sons. Nous apprendrons également l'importance de la numérisation et de la compression de cette information dans le but de stocker, conserver, transmettre et reproduire celle-ci de manière efficace. Ce chapitre a également pour but de présenter la norme MPEG-7, très intéressante dans le cadre de la gestion électronique de documents multimédia, puisqu'elle s'attache à décrire le contenu des documents multimédia, à l'aide de schémas XML.

La création de descripteurs de documents multimédia au format XML permet de réaliser des systèmes offrant des fonctionnalités d'indexation et de recherche de documents audiovisuels. En effet, il suffirait d'associer chaque document audiovisuel à son descripteur et d'indexer les différents éléments de ce dernier pour pouvoir retrouver aisément le document multimédia en question. Dans le domaine des documents multimédias, il existe principalement deux types de recherche: la recherche par mots-clés et la recherche par le contenu. La première approche consiste à interroger le système à l'aide d'une requête constituée de mots-clés, comme c'est le cas dans les moteurs de recherche courants (par exemple, Google ou Alta Vista). Cela correspondrait, dans notre exemple, à effectuer une requête en introduisant les termes "Jacques Brel", pour retrouver toutes les vidéos où l'on peut apercevoir ce dernier. Evidemment, dans ce cas, il faut que toutes les vidéos du système aient été décrites dans un document XML, de façon à indexer les différents éléments du descripteur. De cette façon, il est possible de retrouver les descripteurs de toutes les vidéos concernant Jacques Brel ainsi que, indirectement, les vidéos associées. Le troisième chapitre de ce mémoire a pour finalité de décrire le Data Manager, un système de recherche documentaire multimédia par mots-clés développé par la société Arafox. Ce système, auquel j'ai eu l'occasion de me familiariser durant mon stage de fin d'études, s'inscrit dans une optique Open Source. Il s'articule autour d'une base de données XML native et d'un moteur d'indexation et de recherche que nous évaluerons à travers le

quatrième chapitre. La seconde approche, la recherche basée sur le contenu, permet de récupérer les documents multimédia ayant des caractéristiques visuelles ou auditives données. Ce type de recherche permet, par exemple, d'effectuer des requêtes telles que "retrouvez, s'il vous plait, toutes les images composées de 25% de couleur rouge, 5% de couleur verte et 70% de couleur bleue", ou encore "retrouvez, s'il vous plait, toutes les images contenant une forme rectangulaire dans le coin supérieur droit". Plus intéressant, encore ce type de recherche permet d'effectuer une requête par document-type. Cela consisterait, dans notre exemple, à fournir une photographie de Jacques Brel pour retrouver les vidéos contenant des personnages similaires. A cette fin, il faudrait que le système dispose d'un outil permettant d'extraire les caractéristiques visuelles des images constituant les vidéos du système. Ces caractéristiques seraient alors stockées pour permettre leur comparaison avec les caractéristiques de l'image-type. De cette manière, le système renverrait les vidéos comportant des images ayant les mêmes caractéristiques que l'image donnée. Nous verrons que le Data Manager, tel qu'il est décrit dans le troisième chapitre ne permet pas ce type de recherche, puisque les descripteurs XML de ce système concernent des caractéristiques hors contenu (par exemple, l'auteur du document, sa date de création, ou son type) ou des caractéristiques subjectives (i.e. une description manuelle du document). C'est pourquoi nous verrons, dans le cinquième chapitre de ce mémoire, comment améliorer le Data Manager en nous tournant vers des techniques d'indexation et de recherche basées sur le contenu. Nous aborderons donc, dans un premier temps, le problème de l'extraction des caractéristiques d'un contenu multimédia. Nous verrons que certaines d'entres elles (par exemple, les couleurs, les textures ou les formes) peuvent être extraites de manière automatique alors que d'autres (par exemple, les actions représentées dans l'image) nécessitent une intervention extérieure. Ce chapitre permet également d'insister sur l'intérêt et l'importance de la norme MPEG-7 pour l'indexation et la recherche des documents audiovisuels par le contenu, à travers l'exemple de deux projets.

Ensuite, dans le sixième chapitre de ce mémoire, nous nous pencherons sur le cas particulier des systèmes de recherche d'images basée sur le contenu. Nous étudierons donc, dans un premier temps, le fonctionnement général d'un tel système, dans lequel les images peuvent être retrouvées sur base des couleurs, formes, textures ou encore les visages qu'elles contiennent. Nous nous intéresserons ensuite, plus particulièrement, à la recherche d'images basée sur la reconnaissance de visages.

Partie 1

Concepts

CHAPITRE PREMIER - LA GESTION ELECTRONIQUE DE DOCUMENTS

I.1 – La gestion documentaire manuelle

Afin d'introduire la problématique liée à la gestion documentaire manuelle dans une entreprise, prenons l'exemple d'un client ayant perdu une facture et désirant en recevoir une copie. La secrétaire, après avoir noté les références de la facture et les coordonnées du client, doit se rendre dans le local d'archives où elle recherche le document demandé. Une fois celui-ci trouvé, il sera photocopié avant d'être reclassé dans le local d'archives. La secrétaire peut alors envoyer la copie (par fax ou par courrier) au client. Elle pourra enfin revenir à son bureau et attendre la réponse du client. La secrétaire aura perdu ainsi une grande partie de son temps à rechercher le document, même si celui-ci était bien classé.

Supposons maintenant qu'au même moment, le service de comptabilité de l'entreprise désire retrouver toutes les factures de ce mois. Parmi ces factures se trouve celle de notre client distrait. Le comptable se rend au local d'archives, peu avant la secrétaire, pour récupérer toutes les pièces comptables dont il a besoin et les emporte à son bureau. Dès lors, la secrétaire qui se rendra au local d'archives ne trouvera pas le document qu'elle recherche. Elle devra donc interroger tous les services de l'entreprise afin de récupérer la facture et continuer son travail. Comme l'utilisateur l'observe aisément sur cet exemple, *les nombreuses manipulations des documents dans une entreprise entraînent une perte de temps considérable*.

Admettons à présent que le comptable perde une facture en chemin, entre le local d'archive et son bureau. Le service de nettoyage passant par-là pourrait jeter le papier à la poubelle, par inadvertance, rendant celui-ci à jamais inaccessible. Ainsi, *un document étant sans cesse déplacé de service en service risque de disparaître ou de se dégrader*.

Au fil des ans, l'entreprise amassera de plus en plus de documents qu'il faudra stocker et classer. Dans une dizaine d'années, elle devra ainsi construire une annexe, déménager ou détruire les plus anciens documents afin de trouver la place pour entreposer les plus récents. Les documents prennent de la place et les locaux d'archives d'une entreprise ne sont pas extensibles à souhait. De plus, un document appelé à être conservé demande, en fonction de sa nature, des conditions particulières de chauffage, d'éclairage, d'humidité, etc. Une entreprise ne peut donc pas décider d'entreposer une partie de ses archives au sous-sol, sans un minimum d'installation. *Le stockage de documents physiques coûte toujours cher*.

Supposons que l'entreprise engage un nouveau responsable pour le service d'archivage. Celui-ci décide que, pour des raisons de facilité de recherche de documents, il serait plus pertinent de changer de politique de classement. Ainsi, au lieu d'être classés par ordre chronologique, puis par ordre alphabétique, les documents seront classés en fonction de leur nature puis, par ordre alphabétique, et enfin par ordre chronologique. Le service d'archivage passera alors plusieurs semaines à réorganiser les archives. De plus, pendant cette période, certains documents seront classés selon la nouvelle organisation, certains seront toujours classés à l'ancienne, et certains seront en cours de classement. Il sera donc

extrêmement difficile de retrouver un document ce qui entraînera un retard considérable dans l'un ou l'autre service.

Enfin, une fois la réorganisation achevée, la période d'adaptation de certains employés pourrait être longue. Ayant toujours connu l'ancien classement, le personnel de l'entreprise aura attrapé des automatismes dans la consultation des archives, et éprouvera des difficultés à changer son mode de recherche. *La réorganisation des documents peut avoir pour conséquence une perte de temps considérable ainsi qu'un mécontentement des intéressés.*

Admettons que l'entreprise décide de donner la possibilité à ses clients de la contacter également par courrier électronique. Le personnel responsable de la gestion du courrier devra donc s'adapter en conséquence. Une solution est qu'il gère le courrier électronique, après l'avoir imprimé, de la même façon que le courrier traditionnel. Dans ce cas, l'entreprise ne gagne pratiquement rien à proposer le contact par e-mail et y perd plutôt en termes de coûts d'impression. Une autre solution est d'informatiser la gestion du courrier électronique tout en continuant à gérer le courrier traditionnel de la même manière. Dans ce cas, il faudra jongler entre deux modes de gestion de courrier, avec le risque de devoir engager du nouveau personnel pour répartir les deux types de traitement. De plus, lorsque l'on devra retrouver le courrier d'un client, il faudra non seulement rechercher dans le local d'archives mais également interroger la base de données relative aux courriers électroniques.

Comme le montre cet exemple, les documents physiques demandent beaucoup de place ainsi que des installations particulières afin d'optimiser leur durée de vie, ce qui coûte cher à l'entreprise. De plus, la manipulation de tels documents entraîne une lenteur d'exécution et un risque de perte d'information. Sans compter que le système de gestion manuelle de documents s'adapte mal aux nouveaux supports d'information qui ont de plus en plus un caractère numérique. La numérisation des documents peut faciliter le stockage, le classement, la recherche et la réutilisation des données. C'est pourquoi, il est impératif d'abandonner le caractère physique des documents pour un caractère numérique et passer à une gestion électronique de documents.
Evidemment, comme nous le verrons plus loin, le numérique montre également des inconvénients (par exemple, le problème d'authenticité du document) qu'il faut prendre en compte lorsque l'on désire gérer des documents de manière électronique.

I.2 – Du document physique au document numérique

Dans un souci de gain de place et de sécurité des archives, et du fait de l'augmentation croissante des systèmes informatisés ainsi que des documents électroniques associés, nous assistons de plus en plus à une dématérialisation de la donnée papier par la numérisation.

I.2.1 – Le document

Selon les premières définitions, un document est décrit comme un renseignement écrit servant de preuve ou d'information. Petit à petit, par extension, le document a pris également la forme d'un objet pouvant apporter un renseignement, établir ou infirmer un fait. Ensuite, la signification du mot document a commencé à varier en fonction des domaines

auxquels il s'appliquait. Ainsi, par exemple, pour la gestion de document et la science de l'information, il s'agit de données rassemblées sur n'importe quel support (i.e. papier ou électronique), destinées à la consultation, à l'étude ou à servir de preuve. Un autre exemple est celui du domaine de l'application informatique, où le document est vu comme un écrit ou un enregistrement électronique associé à une activité productrice ou consommatrice d'information. Le document de l'informatique, quant à lui, est vu comme un fichier créé par un utilisateur via un logiciel d'application (par exemple, un texte écrit grâce à un logiciel de traitement de texte ou un graphique réalisé à l'aide d'un tableur) et enregistré sur un support de stockage.
Dans ce domaine, le document étant le type de fichier le plus fréquemment rencontré et donc le plus familier à l'utilisateur, il n'est pas rare de confondre les termes fichier et document.

Quelque soit la définition du terme, il apparaît toujours que le document est un support d'information. La notion de support est extrêmement importante car tant que l'information n'est pas représentée sur ou dans un objet matériel, le document n'existe pas.
Ne dit on pas *"les paroles s'envolent et les écrits restent"*? Cet adage nous montre que si l'on ne garde pas trace d'une information, celle-ci disparaît et l'on ne peut plus prouver qu'elle a un jour existé. Ainsi, par exemple, si l'on diffuse un message en direct à la radio sans enregistrer celui-ci, il sera impossible de le réutiliser par après ou de prouver son existence.

I.2.2 – La conservation du document électronique

Nous avons vu l'importance de passer d'une culture papier à une culture électronique, notamment dans le cas d'une entreprise. La gestion électronique de documents est génératrice d'un nombre extrêmement important de données. Toute cette information est appelée à être consultée pendant de nombreuses années. Donc, les supports de cette information doivent permettre une conservation à long terme sans modification possible, afin de garantir l'authenticité et l'intégrité du document. Mais comment garantir l'authenticité et l'intégrité du document dans cet environnement digital? Ce point sera discuté dans la section I.2.3.

Lorsqu'une personne consulte un document, elle s'attend à ce que celui-ci soit exact, original. Dans le cas d'un document papier, l'information et le support sur lequel elle est inscrite sont indissociables. Son authenticité est liée à la notion d'original et de copie. Le document original est celui qui provient véritablement de l'auteur auquel on l'attribue. Mais dans le cas du numérique, un document peut facilement être modifié, corrigé ou même détruit. De plus, le format électronique permet l'insertion d'annotations, par exemple, au sein même du document. C'est pourquoi, contrairement au papier, le support numérique est jugé non fiable et de courte durée de vie. C'est pour cette raison que l'on préférera utiliser des supports optiques non réinscriptibles de type WORM (Write Once Read Many), car une fois écrit, on ne peut ni modifier ni supprimer le contenu. Mais certains documents sont trop volumineux pour être stockés sur de tels supports. Les images médicales, par exemple, ne peuvent être stockées sur des disques optiques, à moins de subir une compression à très grande échelle. Ceci rendrait alors le document infidèle à l'image originale.

Un autre problème est celui de la lisibilité des documents. Depuis l'apparition des supports virtuels tels que les supports magnétiques, l'homme doit passer par l'intermédiaire d'une machine pour prendre connaissance du contenu d'un document. Mais la rapidité de l'évolution technologique nous empêche d'affirmer que l'on disposera toujours du matériel/logiciel nécessaire à la lecture du document électronique dans quelques années.

Une première solution serait donc de conserver ces matériels et logiciels pendant toute la durée de vie du document. Mais une solution plus réaliste serait de migrer les données à chaque fois que l'on change de système informatique.

I.2.3 – Les problèmes juridiques liés au document électronique

Le document électronique est sujet à des problèmes juridiques liés à son caractère immatériel. Lorsque l'on consulte un document papier, par exemple, on s'attend à ce que celui-ci soit authentique. Il n'y a aucune raison qu'il en soit autrement pour un document électronique. Toutefois, le problème de l'authenticité d'un document électronique est plus complexe que dans le cas d'un document papier. Cette catégorie de documents est également touchée par d'autres problèmes juridiques tels que l'intégrité, la confidentialité ou encore les problèmes de droit d'auteurs.

A – Les problèmes d'intégrité, d'authenticité et de confidentialité

Le plus souvent, un document électronique est appelé à voyager dans le monde entier, à travers les réseaux. Admettons par exemple qu'Alice veuille envoyer un document à Bob. Comment s'assurer que ce document n'est pas intercepté par une tierce personne qui en détournerait les données à des fins illicites? Comment Bob peut-il être sûr que le document qu'il reçoit correspond bien à celui qu'Alice lui a envoyé?
En plus de ce problème d'intégrité du document, Bob doit également pouvoir s'assurer de son authenticité. Lorsqu'il s'agit d'un support papier, l'authenticité du document est reconnue s'il est signé de la main de l'expéditeur. Mais qu'en est-il du document électronique?
Enfin, un dernier souci pour Bob serait de s'assurer qu'il est le seul à pouvoir lire les données se trouvant sur le document, dans le cas où il contiendrait de l'information confidentielle.

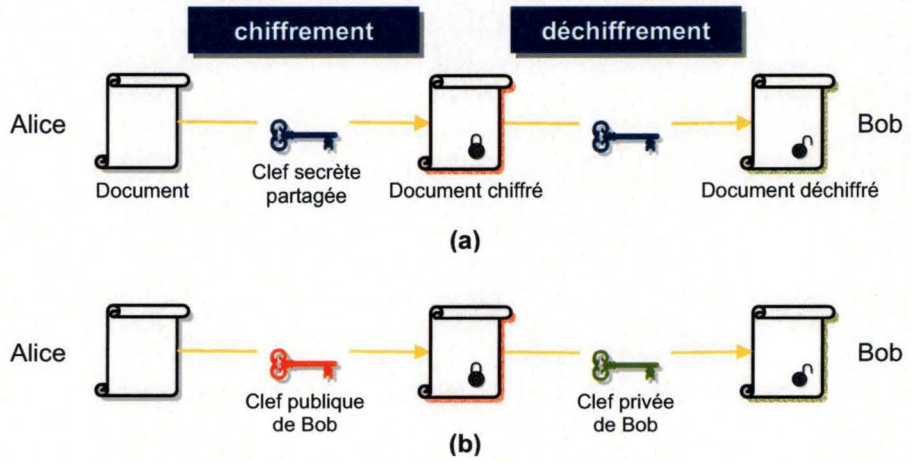


Figure I.1. Principe de la cryptographie. **(a)** Cryptographie symétrique. **(b)** Cryptographie asymétrique.

La réponse à ces problèmes est apportée par la cryptographie qui est, en quelque sorte, l'art de rendre un message incompréhensible. De manière générale, le chiffrement d'un document consiste à coder celui-ci à l'aide d'un algorithme de chiffrement auquel on applique une clef. Pour être lu, le document devra par la suite être décodé à l'aide d'une clef. Comme le montre la figure I.1, la méthode de chiffrement utilisée peut être de deux types.

Dans le cas de la cryptographie symétrique, Alice et Bob se partagent la même clef secrète (la clef bleue, dans la figure I.1 (a)). Chaque clef secrète est propre au couple (émetteur, destinataire). Donc, pour chiffrer le document qu'elle désire envoyer à Bob, Alice applique la clef secrète. Le document ainsi obtenu est codé et n'est lisible que par les propriétaires de la clef secrète, c'est-à-dire Bob et Alice. De cette façon, pour lire le document qu'Alice lui a envoyé, Bob y applique la clef secrète et obtient un document lisible.
Dans le cas de la cryptographie asymétrique, Bob et Alice possèdent tous les deux une clef privée et une clef publique qui est une fonction irréversible de la clef privée. A la figure I.1 (b), la clef privée de Bob est représentée en vert et sa clef publique en rouge. Ici, Alice chiffre le document à l'aide de la clef publique de Bob. Il en résultera un document que seul Bob pourra lire. Si ce dernier désire déchiffrer le document, il lui suffira d'y appliquer sa clef privée.
Etant donné que la clef secrète est propre au couple d'acteurs et qu'il n'est pas possible que deux acteurs différents aient la même paire (clef publique, clef secrète), on peut être certain qu'un document chiffré par Alice ne pourra être lu que par Bob.

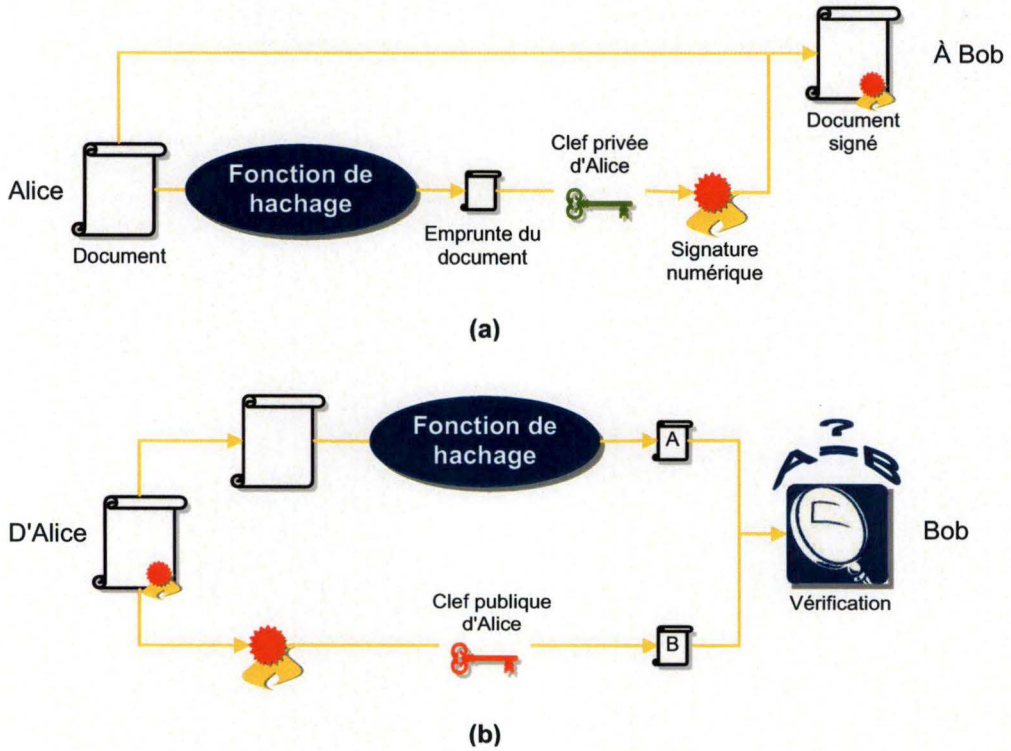


Figure I.2. Principe de la signature électronique. **(a)** Signature du document par l'émetteur. **(b)** Vérification de l'authentification de l'émetteur par le destinataire.

La cryptographie n'assure pas seulement l'intégrité et la confidentialité du document; elle peut également en assurer l'authenticité. En effet, avec la cryptographie, il est possible de signer un document électronique, de manière numérique. La figure I.2, inspirée de [MON04], illustre le principe de signature électronique. Dans un premier temps, comme le montre la figure I.2 (a), Alice va signer le document qu'elle désire envoyer à Bob. A cette fin, le document va subir une fonction de hachage qui fournira une sorte de résumé du message, appelé empreinte numérique. C'est sur ce résumé que va être appliquée la clef privée d'Alice (représentée en vert dans l'illustration). Il en résultera un document crypté qui représente la

signature d'Alice[1]. Alice pourra donc envoyer le document et la signature à Bob[2]. Dans un second temps, Bob qui aura reçu le document signé, désirera s'assurer de son authenticité (cf. figure I.2 (b)). Pour cela le document subira une fonction de hachage fournissant comme résultat une empreinte A. Parallèlement à cela, la clef publique d'Alice (en rouge dans l'illustration) sera appliquée à la signature électronique du document. Il en résultera un document B, similaire à l'empreinte A. Pour vérifier l'authenticité du document, il suffira donc de comparer A et B entre eux. S'ils sont égaux, le document est authentique, sinon, il ne l'est pas.
Il est à noter qu'un prestataire de service de certification assure que la paire de clefs d'Alice lui appartient[3].
Notons également qu'il est tout à fait possible pour Alice d'envoyer à Bob un message confidentiel chiffré et signé numériquement. Elle utilisera sa clef privée pour signer le document et la clef publique de Bob pour le chiffrer. Bob, à son tour, utilisera sa propre clef privée pour déchiffrer le document et la clef publique d'Alice pour en vérifier l'authenticité.

Bien entendu, la cryptographie ne peut offrir de solutions aux problèmes d'authenticité que pour les documents nés électroniques (en opposition avec les documents papier numérisés). En effet, la signature électronique ne permet en aucun cas de prouver que le document électronique correspond bien à la version papier, ni même l'authenticité du document initial (i.e. la version papier). Toutefois, les solutions offertes par la cryptographie sont suffisantes dans le cadre de ce mémoire, puisqu'elles permettent d'authentifier les documents multimédia.

B – Le problème de la preuve des actes juridiques dans l'environnement électronique

A présent que nous savons que la signature électronique d'un document permet d'en vérifier l'authenticité, nous pourrions nous demander si ce document électronique signé numériquement permet de faire une preuve.
En droit belge, les paragraphes 4 et 5 de l'article 4 de la loi du 9 juillet 2001[4] définissent les effets juridiques des signatures électroniques:

> ***Art. 4.*** *§ 4. « Sans préjudice des articles 1323 et suivants du Code civil, une signature électronique avancée réalisée sur la base d'un certificat qualifié et conçue au moyen d'un dispositif sécurisé de création de signature électronique, est assimilée à une signature manuscrite, qu'elle soit réalisée par une personne physique ou morale. »*
>
> *§ 5. « Une signature électronique ne peut être privée de son efficacité juridique et ne peut être refusée comme preuve en justice au seul motif:*
>
> – *que la signature se présente sous forme électronique, ou*
> – *qu'elle ne repose pas sur un certificat qualifié, ou*
> – *qu'elle ne repose pas sur un certificat qualifié délivré par un prestataire accrédité de service de certification, ou*
> – *qu'elle n'est pas créée par un dispositif sécurisé de création de signature. »*

[1] Etant donné que la signature numérique est obtenue par transformation du document, ces derniers sont liés par un lien logique qui peut rappeler le lien physique qui existe entre un document papier et une signature manuscrite.
[2] L'émetteur envoie bien deux fichiers au destinataire, l'un étant le document et l'autre représentant la signature électronique.
[3] La phase d'obtention d'un certificat passe par la génération d'une paire de clefs, la vérification de l'identité de l'acteur, la génération du certificat et enfin le stockage de celui-ci dans un registre électronique. De cette façon, n'importe qui peut obtenir la clef publique de l'émetteur pour vérifier la signature du document.
[4] Loi fixant certaines règles relatives au cadre juridique pour les signatures électroniques et les services de certification.

Le paragraphe 4 transpose en droit belge le principe d'assimilation. Cela signifie que les documents électroniques liés à une signature électronique qualifiée[5] sont assimilés à des documents papier signés manuscritement. Donc, non seulement ils doivent être pris en considération par le juge, mais en plus, ils ont force probante[6]. Le paragraphe 5 transpose le principe de non-discrimination qui signifie qu'un document numérique lié à une signature numérique doit être examiné par le juge.
L'article 1322, alinéa 2 du Code civil quant à lui fournit une définition fonctionnelle de la signature électronique:

> ***Art. 1322***. *« L'acte sous seing privé, reconnu par celui auquel on l'oppose, ou légalement tenu pour reconnu, a, entre ceux qui l'on souscrit et entre leurs héritiers et ayants cause, la même foi que l'acte authentique. »*
> *« (Peut satisfaire à l'exigence d'une signature, pour l'application du présent article un ensemble de données électroniques pouvant être* ***imputé*** *à une personne déterminée et établissant le* ***maintien de l'intégrité*** *du contenu de l'acte.) »*

En utilisant le terme "imputé", les législateurs renvoient aux fonctions d'identification et d'adhésion au contenu que doit avoir une signature classique. Cela signifie que l'ensemble des données électroniques doit permettre d'identifier l'individu ou d'affirmer son adhésion au contenu pour être considéré comme une signature électronique. En outre, il est question ici d'une troisième fonction, propre au caractère électronique de l'acte. Cela signifie que la signature doit pouvoir assurer l'intégrité du contenu du document. Donc, si les fonctions d'imputabilité et d'intégrité sont assurées, le document électronique est assimilé à un acte sous seing privé.

C – Le droit d'auteur

Le caractère numérique d'un document facilite sa reproduction (et donc l'exploitation de l'œuvre d'autrui), la célèbre fonction *copier/coller* nous le prouve. Toutefois, il faut savoir que tout n'est pas permis: les auteurs des documents ont des droits.
Selon [DGLM03], le droit d'auteur *"confère aux auteurs des droits exclusifs relatifs à l'utilisation de leur œuvre"*. Donc, pour pouvoir utiliser une œuvre, il faut y avoir été autorisé par le titulaire de droits sur celle-ci. Les œuvres en question peuvent être:

- Des **textes** de toute nature, quels que soient leur contenu, leur longueur, leur destination ou leur forme.
- Des **photographies** quels que soient leur support et leur objet.
- Des **images** qu'elles soient virtuelles ou non et quel que soit leur type.
- Des **séquences musicales**, **vidéo** ou **audiovisuelles** quel que soit le format ou le support d'enregistrement.
- Des **programmes d'ordinateur**.

Les droits d'auteur sont de deux types:

- Les **droits patrimoniaux** qui permettent à l'auteur de retirer un bénéfice économique de l'exploitation de son œuvre. Ils reprennent le *droit de reproduction*, le *droit d'autoriser l'adaptation et la traduction de l'œuvre*, le *droit de location ou de prêt*, le *droit de distribution* et le *droit de communication au public*.

---

[5] Il faut entendre par signature électronique "qualifiée", une signature électronique avancée, basée sur un certificat qualifié (i.e. fourni par un prestataire de service de certification) et créée par un dispositif sécurisé de création de signature.
[6] Selon [MON04], la force probante est *"l'intensité quant à la preuve que la loi reconnaît à un élément de preuve et qui s'impose au juge"*.

- Les **droits moraux** qui visent la protection de l'intégrité de l'œuvre, la relation de celle-ci avec son auteur et la réputation de l'auteur. Ils reprennent le *droit de divulgation*, le *droit de paternité* et le *droit à l'intégrité*.

A présent que nous avons vu ce qu'était un document, nous pouvons nous pencher sur la *gestion électronique de documents*, qui sera l'objet de la prochaine section.
Notons que dans ce travail, on parlera de *document papier* ou encore de *document physique* par opposition au *document électronique*, *numérique* ou *digital*.

I.3 – Les systèmes de gestion électronique de documents

La gestion électronique de documents (ou GED) consiste en la consultation d'information ou de documents électroniques qui doivent être stockés en vue de pouvoir être récupérés. Elle a donc pour objectifs d'acquérir des documents, de les mémoriser et de permettre de les retrouver pour les diffuser.
Dans un premier temps, les solutions de la GED s'appliquaient exclusivement aux documents papier qui étaient numérisés. A cela sont venus s'ajouter la prise en charge des spools informatiques (documents issus de l'informatique), et celle des documents issus directement d'un traitement bureautique.

I.3.1 – La numérisation des documents

Le point d'entrée d'un système de gestion électronique de documents est l'obtention d'un document sous forme électronique, sans quoi il ne pourrait pas être traité par des outils informatiques. Or, même si à l'heure actuelle la transmission de documents numérisés est de plus en plus présente au sein de l'entreprise, les documents papier y circulent toujours en très grand nombre. C'est pourquoi, il est souvent nécessaire de numériser ces données "papier" afin de les faire entrer dans un système de GED.

Dans le cas d'un document préexistant, il faut le dématérialiser ou le convertir sous forme numérique. La numérisation du document dépend de son type. Ainsi, par exemple, un document textuel peut être numérisé soit en mode "caractère", soit en mode "page". Dans le premier mode, les éléments du texte pourront être accessibles alors que dans le second mode, la numérisation résultera en une "photographie" du document dans laquelle les éléments du texte ne seront pas manipulables. Donc, si l'on souhaite traiter l'information contenue dans le document initial, il faudra passer par une phase d'OCR[7] après la première phase de numérisation.

Dans le cas d'un document nouveau, il suffit de le créer directement sous forme numérique sans passer par un support intermédiaire. Ici aussi, la création du document numérique dépendra du type de celui ci. Le document sera alors purement virtuel pour l'utilisateur et ne se matérialisera que lors d'une édition sur un support traditionnel.
La création du document textuel, structuré ou non, se fait grâce à une saisie manuelle au clavier.

[7] OCR = Optical Character Recognition.

I.3.2 – L'organisation et les fonctions d'un système GED

Les premières solutions GED, s'inscrivant dans la lignée des systèmes d'archivage électronique, étaient essentiellement autonomes ou reliées à des réseaux locaux.
Ces systèmes visaient la suppression des salles d'archives en remplaçant le document initial par une copie numérique de celui-ci.
Comme l'illustre la figure I.3, issue de [CHA96], une station d'archivage autonome est construite autour de quatre éléments que sont l'ordinateur, le numériseur, l'unité de disque optique magnétique et l'imprimante. Ceux-ci permettent, respectivement, le traitement, l'acquisition le stockage et la restitution des documents.

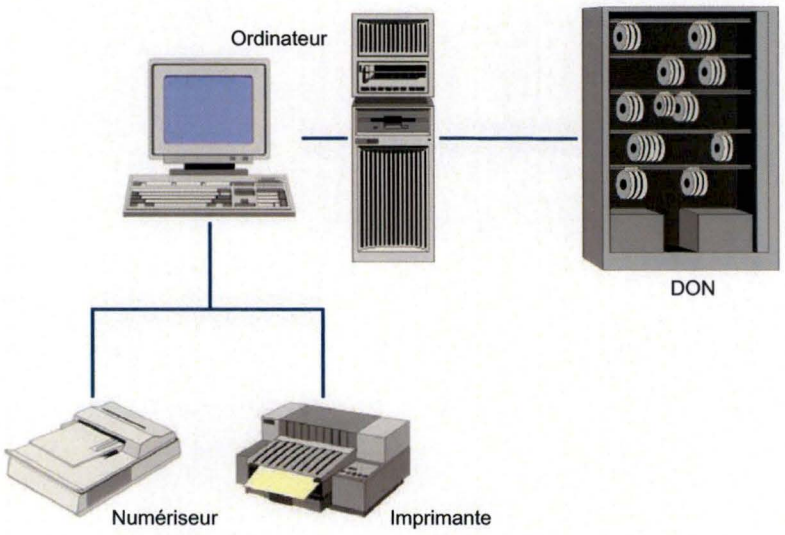


Figure I.3. Station d'archivage autonome.

L'ordinateur gère les documents et les données qui leur sont associées. C'est lui qui gère l'ensemble de la station de travail. Il sert également à indexer les documents, à les rechercher ou encore à les consulter. Le numériseur sert à l'acquisition des documents en mode page ou en mode caractère, s'il comporte un système OCR.
Ce type de station autonome est surtout utilisée dans les applications de type bureautique comme la gestion de courrier.

Un système GED en mode client/serveur permet à un grand nombre d'utilisateurs d'y accéder simultanément. La construction d'un tel système d'archivage se fait grâce à un réseau de communication, certaines composantes matérielles et un logiciel (voir la figure I.4, reprise de [CHA96] et de [PEL99]). Le réseau de communication est l'élément qui va permettre aux utilisateurs d'accéder au système et donc aux documents.
Parmi les composantes matérielles, on compte une station d'acquisition, une unité de stockage, différents serveurs et des postes clients. La station d'acquisition comprend habituellement un numériseur (mode page ou relié à un lecteur OCR). Le stockage des documents dans l'unité de stockage se fait généralement à l'aide d'un juke-box de disques optiques permettant ainsi de stocker un important volume de documents.
A chaque document du système est associée une série de caractéristiques le décrivant (caractéristiques physiques du document et éléments descriptifs de son contenu). Ces caractéristiques vont permettre d'indexer les documents pour leur recherche. Un serveur de base de données permet le traitement de toute l'information concernant les documents. La gestion des éléments physiques des documents se fait via un serveur de documents qui

possède des fonctions de manipulation des documents numérisés telles que la compression et la décompression. La matérialisation des documents se fait via un serveur d'impression. La restitution des documents peut être demandée depuis les différents postes de consultation. Un serveur de fax permet l'envoi automatique des documents sur le fax de l'utilisateur destinataire.
Les clients du système GED sont les différentes stations de travail et de consultation équipées d'ordinateurs. Le traitement d'un document d'un système GED passe par son acquisition, sa restitution, et ensuite sa diffusion.

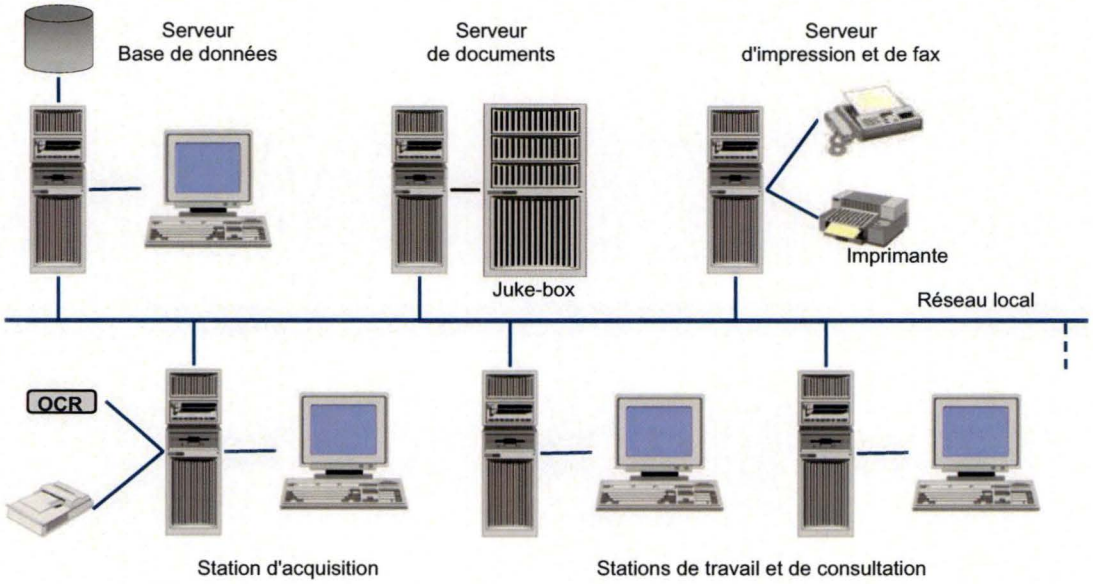


Figure I.4. Station d'archivage en mode client-serveur.

A – L'acquisition des documents

L'acquisition d'un document consiste essentiellement en la création du document électronique (en mode caractère si l'on souhaite disposer de son contenu, en mode page sinon). Dans le cas de la numérisation en mode caractère, le document est également numérisé en mode page. Ainsi, on dispose d'un fichier ASCII reprenant les données et d'une image pour que l'utilisateur puisse voir le document tel qu'il était initialement.

L'orthographe du texte d'un document issu d'une analyse OCR ou d'une création en traitement de texte est corrigée (via un correcteur orthographique). Ainsi, les termes bien orthographiés uniquement pourront être utilisés lors de la recherche de documents.

L'acquisition d'un document passe également par l'acceptation de son format (celui-ci doit être accepté par le système pour être importé) ou la conversion de celui-ci en un autre format (reformatage du document). Le traitement des documents en mode caractère est illustré à la figure I.5 ([CHA96]).

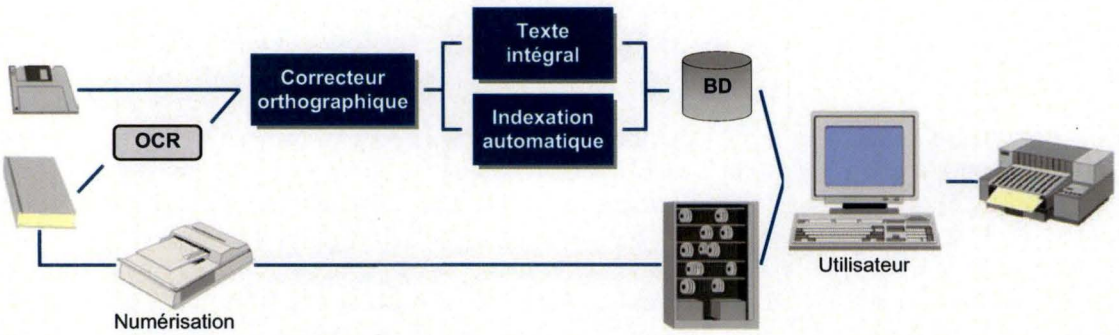


Figure I.5. Traitement de documents en mode caractère.

Dans certains cas, lorsque le document est uniquement numérisé en mode page, l'acquisition de celui-ci passe aussi par une phase d'indexation manuelle (voir la figure I.6 reprise de [CHA96]). Il faudra alors analyser le document pour repérer une série de mots clés représentant son contenu et des éléments descriptifs (titre, auteur, etc.).

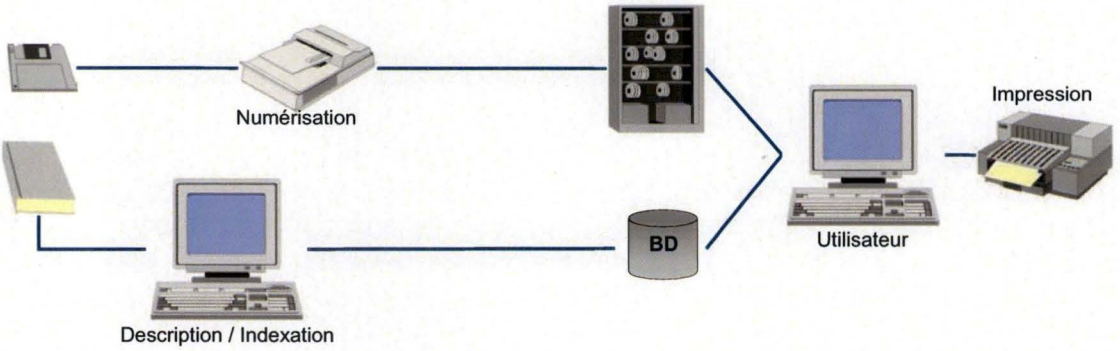


Figure I.6. Traitement de documents en mode page.

Les données qui entrent dans un système GED ne sont pas uniquement des documents papiers que l'on a numérisés. Ils peuvent également provenir de fichiers bureautiques, de formulaires électroniques ou de serveurs de fax.

B – La restitution des documents

La restitution d'un document consiste en la recherche, l'affichage et l'impression de celui-ci.

La recherche

Puisque c'est grâce à elle qu'un utilisateur peut sélectionner les documents dont il a besoin parmi tous ceux du système, la phase de *recherche* constitue la fonctionnalité la plus importante dans l'étape de restitution.
Les systèmes GED peuvent abriter deux types de moteurs de recherche: les moteurs de recherche développés autour d'un SGBDR[8] tel que ORACLE ou MySQL, et ceux développés autour d'un logiciel documentaire. Les premiers moteurs de recherches sont destinés à des

[8] Pour Système de Gestion de Bases de Données Relationnelles.

applications où la recherche est conduite de manière arborescente. Les derniers quant à eux sont utilisés pour leurs fonctions de recherche sur le texte des documents, l'utilisateur peut alors fournir n'importe quel terme du texte comme critère de recherche.

Lorsqu'il effectue une recherche, l'utilisateur n'utilise pas forcément le terme qui est à l'origine de l'indexation du document qu'il désire trouver.
Supposons qu'un utilisateur, désireux de savoir comment la population d'Alaska se nourrit, interroge le moteur de recherche en introduisant les termes "nourriture" et "Alaska". Il reçoit alors, comme résultat, tous les documents contenant ces deux mots ou uniquement l'un des deux. Mais il aurait peut-être eu davantage de documentation s'il avait introduit "alimentation" à la place de "nourriture". D'où l'importance de la prise en charge des synonymes pour la fonction de recherche des documents.
En outre, si cet utilisateur est distrait, il doit pouvoir retrouver l'information qu'il recherche, même s'il a omis un "r" à "nourriture". Dès lors il faudrait également que le texte initial, s'il contient des fautes, soit revu par le correcteur d'orthographe avant d'être indexé. Car si, dans le document, le terme "nourriture" est écrit avec un seul "r", alors il sera indexé en fonction, et l'utilisateur introduisant le terme exact ne le trouvera pas. Ainsi, il est nécessaire de prendre en compte une vérification de l'orthographe lors la recherche et lors de l'indexation du document.

L'affichage et l'impression

Les documents que l'utilisateur sélectionne lors de sa recherche sont affichés par des viewers (programmes qui permettent de visualiser le document). Ces logiciels étant pour la plupart de type WYSIWYG (What You See Is What You Get), le document résultant d'une éventuelle impression sera la matérialisation exacte de l'image affichée à l'aide du viewer. Cette image pourra soit être imprimée, soit être exportée vers un support magnétique.

C – La diffusion des documents

Il est probablement inutile de préciser qu'à l'heure actuelle, la meilleure façon de diffuser un document électronique est d'utiliser la voie des réseaux: Intranet, Extranet et Internet.

I.3.3 – Solutions de la GED

Les solutions de la gestion électronique de document peuvent être de plusieurs types, mais les deux plus grandes catégories sont la GED-COLD et la GED-Image.

La GED-COLD concerne tous les programmes de gestion des données issues de traitements informatiques (factures, bons de livraison). Le mot COLD vient en remplacement des listings qui étaient édités sur des microfiches à partir d'imprimantes COM[9].
Dans ce type d'application, l'entièreté des traitements du document est automatisée. Les fichiers spools générés contiennent des données de mise en page et des données de contenu. Les informations de contenu du document subissent une indexation automatique qui permettra de le retrouver par après. Etant donné que tous les documents d'une application COLD sont très structurés et suivent une présentation fixe, il est facile de retrouver les données de contenu en fonction de leur place sur le document.

[9] COLD vient de Computer Output on Laser Disk et COM vient de Computer Output on Microfilm.

L'information concernant la mise en page peut être stockée une fois pour toutes pour un type de document puisque toutes les factures, par exemple, se présenteront de la même manière. Lorsque l'utilisateur affichera le document via le viewer, ces données seront récupérées pour la présentation des données.

La GED-image reprend tous les programmes de la GED concernant les images ou les documents numérisés.
Notons qu'il n'est pas impératif de dissocier la GED-COLD et la GED-Image parce que la plupart des logiciels de la GED sont capable de gérer tant les fichiers issus de traitements informatiques que les images numérisées.

I.3.4 – Conclusion

Aujourd'hui, grâce aux systèmes de gestion électronique de documents, on peut accéder instantanément à une information, de n'importe quel endroit dans le monde.
L'impact de tels systèmes est bien sûr une augmentation de la productivité. Aussi, le stockage sur des supports optiques permet des gains de place et de manipulation: il est plus facile de dupliquer 10000 factures sur un CD-ROM que de les photocopier.
La GED entraîne également une augmentation de la réactivité au sein de l'entreprise. En effet, puisque l'on accède plus rapidement à l'information, les acteurs de l'entreprise disposent plus rapidement des documents. Cela leur permet de prendre plus rapidement des décisions.
Enfin, la Gestion électronique de documents permet de communiquer avec les autres entreprises et avec les clients.

CHAPITRE DEUXIEME - LES DOCUMENTS MULTIMEDIA

II.1 – Le multimédia

Il y a une dizaine d'années, lors de la journée d'information sur le multimédia[1], Eddy Goray, chef de service de la RTBF, se penchait sur la définition du multimédia.
L'étymologie de ce mot nous indique que le multimédia est tout ce

> *« Qui utilise ou concerne plusieurs médias, c'est-à-dire plusieurs supports de diffusion de l'information. Chaque support de diffusion étant caractérisé par sa forme d'expression et de présentation de l'information du message. »*

De manière similaire, le terme monomédia est utilisé, lorsque l'on fait référence à *un seul support de diffusion de l'information.* Dès lors, Eddy Goray a défini l'information multimédia comme étant *une information composée de plusieurs formes différentes d'information et de supports d'information.* C'est-à-dire, *une information composée de plusieurs monomédia* ([GOR95]).

Mais, ainsi qu'il le faisait remarquer lui-même il y a dix ans, cette définition ne prend pas en compte le lien qui existe entre l'information et son support. C'est ainsi qu'Eddy Goray a proposé de définir le multimédia comme étant une

> *« Association de plusieurs monomédia avec description des liens logiques qui lient les éléments d'information monomédia. »*

L'équipement multimédia peut être défini comme étant un

> *« Équipement susceptible de stocker, conserver, transmettre, reproduire les différents monomédia constitutifs en respectant l'organisation des liens logiques qui les lient. »*

II.2 – L'information multimédia

L'information dont il est fait référence dans la définition du multimédia ci-dessus peut prendre la forme d'une information auditive (i.e. le son) ou visuelle. Ce deuxième type d'information concerne, d'une part, l'information visuelle fixe (i.e. le texte ou l'image) et, d'autre part, l'information visuelle animée (i.e. la vidéo).

[1] Lors de son 25ème anniversaire, l'Institut d'Informatique des Facultés Universitaires Notre-Dame de la Paix, Namur a organisé diverses manifestations à caractère scientifique. L'une d'entre-elles était une journée ayant pour thème le multimédia.

II.2.1 – Les différents types d'information

A – Le texte

L'être humain est capable de tirer de l'information à partir d'une suite de mots constituant un texte. Ceci, bien entendu, à condition qu'il ait appris à reconnaître ces caractères et qu'il dispose d'outils (par exemple un dictionnaire) lui permettant de donner un sens aux mots qu'il lit. C'est ainsi que l'on peut, par exemple, être au courant de l'actualité en lisant son journal, le matin. Bien sûr, il est impossible de lire cette information si elle n'a pas été écrite auparavant. En effet, un groupe de rédacteurs se cache derrière les articles que l'on lit.
La figure II.1 (a) illustre le processus de transmission de l'information textuelle. Premièrement, le rédacteur s'imagine ou visualise un fait (ici, il s'agit d'une voiture de police) qu'il retranscrit dans un article, par exemple, lors d'une phase d'encodage. Ensuite le lecteur, disposant de cet article (ou d'une copie) le déchiffrera lors d'une phase de décodage. Il pourra donc s'imaginer le fait à son tour.

B – L'image

L'image est une autre forme d'information visuelle fixe. L'homme est capable d'interpréter ce qu'il voit sur une image.
La figure II.1 (b) illustre le processus de transmission de l'information contenue dans une image. Dans un premier temps, un photographe, par exemple, assiste à une scène qu'il désire immortaliser (ici, il s'agit d'une voiture de police). La phase d'encodage correspond alors à la photographie de cette scène. Il en résultera une image que l'on pourra consulter. Ensuite, la phase de décodage correspond à l'interprétation que le consulteur fait de l'image.

Souvent, une image accompagne un texte qu'elle illustre (c'est le cas, par exemple des photos accompagnant les articles de presse écrite). L'association de l'image et du texte permet d'imaginer plus facilement l'information que le rédacteur ou le photographe a voulu nous transmettre.

C – Le son

Le son, quant à lui, est une information auditive. L'être humain a la capacité de reconnaître des sons et de leur donner un sens. Ainsi, il est également possible pour l'homme de se tenir au courant de l'actualité en écoutant l'information à la radio. Il est capable de reconnaître les mots qui sont prononcés et d'en deviner la signification, en se l'imaginant.
La figure II.1 (c) illustre le processus de transmission de l'information sonore. Premièrement un narrateur s'imagine ou visualise un fait qu'il enregistre[2], par exemple, lors d'une phase d'encodage de l'information. Par la suite, un auditeur ayant accès à l'enregistrement pourra s'imaginer le fait lors de la phase de décodage.

D – La vidéo

La vidéo est une information visuelle animée. Il s'agit en effet d'une suite d'images fixes qui défilent, à intervalles réguliers.

[2] L'information pourrait ne pas être enregistrée sur un support, mais il est préférable qu'elle le soit (cf. chapitre 1).

La figure II.1 (d) illustre le processus de transmission de l'information contenue dans une vidéo. Dans un premier temps, un caméraman filme une scène (ici, il s'agit d'une voiture de police roulant sur une autoroute). Il s'agit de la phase d'encodage de l'information. Il en résultera une bande vidéo que l'on pourra consulter. Ensuite, lors de la phase de décodage le téléspectateur interprète les images qui défilent.

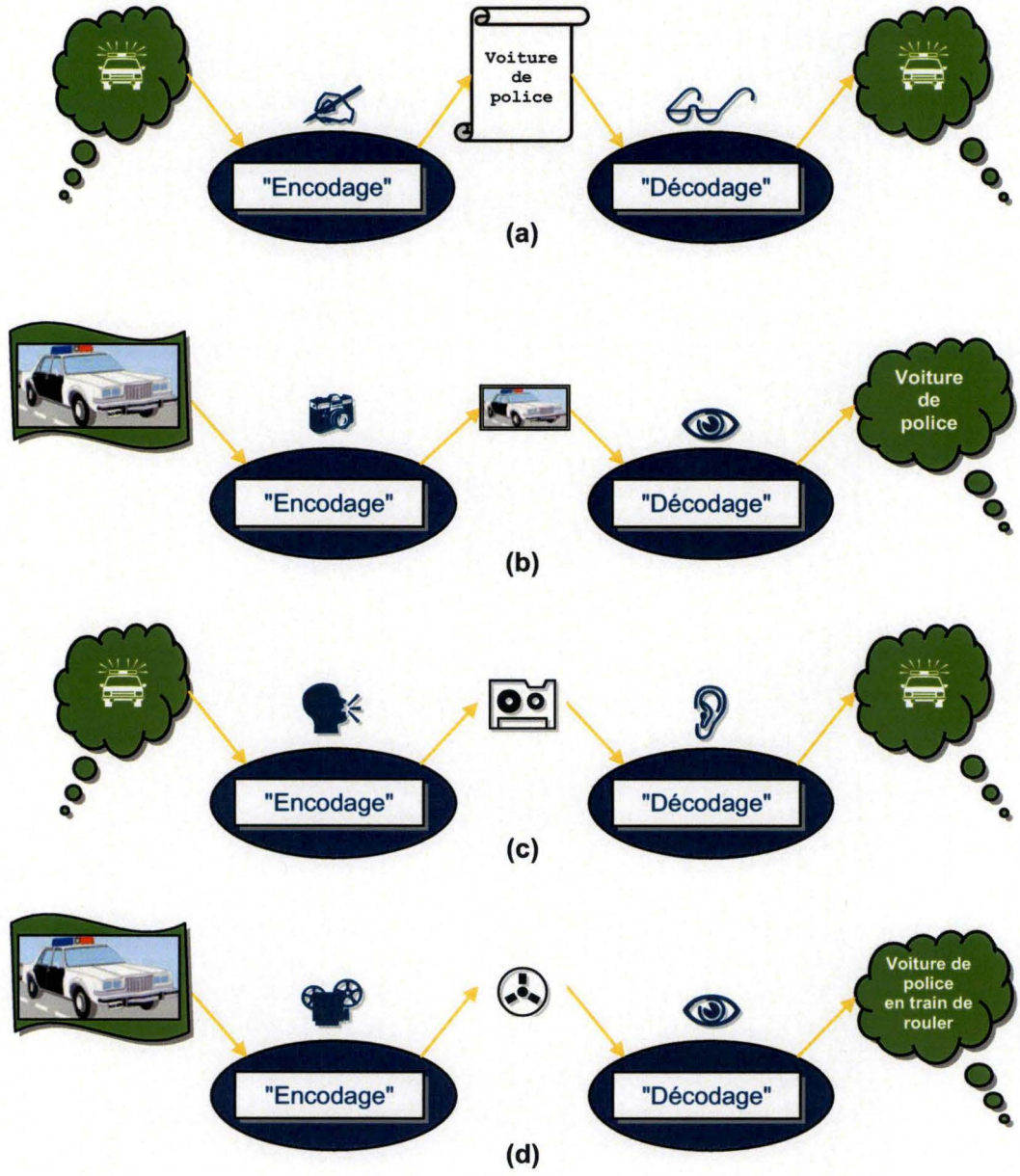


Figure II.1. Processus de transmission de l'information **(a)** textuelle, **(b)** contenue dans une image, **(c)** sonore ou **(d)** contenue dans une vidéo.

Le stockage, la conservation, la transmission et la reproduction des informations en respectant le "scénario multimédia"[3] nécessitent de disposer des différentes informations sous un même format. Ainsi, le multimédia a pu prendre sa place dans notre société grâce à la numérisation et la compression de cette information.

[3] Eddy Goray définit l'organisation de l'ensemble des liens avec les monomédia comme le "scénario multimédia".

II.2.2 – La numérisation de l'information

La numérisation ou digitalisation permet de représenter une information analogique à l'aide d'une suite de nombres binaires.

L'analogique et le numérique

L'analogique et le numérique sont deux moyens pour le transport et le stockage de données. Un support analogique permet la reproduction des valeurs d'un phénomène continu sous une forme similaire. Il s'agit, par exemple, d'une cassette audio ou d'un disque vinyle. De cette façon, lorsque l'on enregistre un son sur une cassette audio, le signal reproduit sur la bande aura la même allure (les mêmes amplitudes) que l'onde sonore. Celui-ci peut être représenté par une courbe. Un signal numérique, quant à lui, ne peut prendre que des valeurs bien définies et pourra donc être présenté par un histogramme.
A la figure II.2, la courbe bleue illustre la représentation d'un signal sous forme analogique, alors que l'histogramme en vert représente le même signal sous forme numérique.

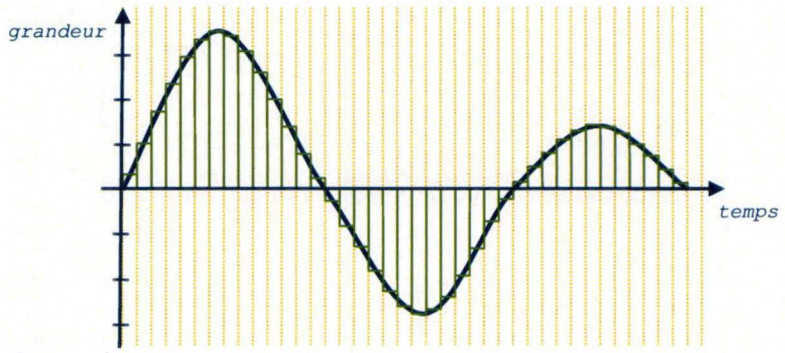


Figure II.2. Représentation du signal sous formes analogique et numérique.

Il en ressort donc que le signal numérique est plus facilement éditable et reproductible que le signal analogique. En effet, supposons que monsieur K. Sète désire copier une cassette audio pour sa sœur. Le résultat de la copie sera un document sonore de moins bonne qualité que le document original. De même que les copies réalisées par la sœur de monsieur K. Sète au départ de la copie reçue de celui-ci seront d'une qualité encore inférieure. En revanche, la copie du compact disque réalisée par monsieur C. Dérom sera, bit pour bit, identique à l'original. Théoriquement, le compact disque pourrait être dupliqué une infinité de fois sans que le contenu s'en trouve altéré.
De plus, le fait de disposer de l'information sous format numérique permet de traiter et d'analyser celle-ci, ce qui n'est pas possible avec l'analogique. Par exemple, lorsque l'on possède une image sous format numérique, on peut l'améliorer, la restaurer, y détecter des formes, etc. C'est pourquoi il est intéressant de numériser, c'est-à-dire de transformer le signal analogique en un signal numérique.

La numérisation

La numérisation consiste en deux phases que sont l'échantillonnage et la quantification. Lors de la première phase, le signal analogique est découpé en tranches temporelles de même épaisseur (i.e. échantillons) auxquelles seront attribuées des valeurs numériques dans la seconde phase. Ces valeurs correspondent à la grandeur (i.e.

l'amplitude du signal) convertie selon une certaine résolution. Cette dernière traduit le nombre de niveaux de la conversion, c'est-à-dire le nombre maximum de valeurs différentes qu'un échantillon peut prendre.

Il est évident que plus les périodes de temps entre lesquelles les échantillons sont prélevés sont petites (i.e. plus les tranches sont fines), plus le signal numérique sera proche du signal analogique. De même, la qualité sera d'autant meilleure que la résolution sera grande.

Tableau II.1. Ce que peuvent contenir 100 Mo selon les différents types de données.

Type de données	Contenu de 100 Mo
Texte brut	100 millions de caractères
Texte avec style	30 millions de caractères
Vidéo non compressée	5 secondes
Vidéo compressée	8 minutes
Son stéréo non compressé	9 minutes 30 secondes
Image 13" noir et blanc	325 images
Image 13" 16 millions de couleurs	14 images

L'inconvénient du signal numérique par rapport à l'analogique est qu'il est beaucoup plus encombrant. La taille des fichiers résultant de la numérisation demande une capacité de stockage et de la puissance. A l'heure actuelle, les micro-ordinateurs ont une capacité de l'ordre de 120 Go et un processeur de l'ordre de 3 Ghz.
A titre d'exemple, le tableau II.1, issu de [HER94], illustre l'espace disque occupé par rapport au type d'information numérisée. On peut constater qu'un CD-Rom d'une capacité de 800 Mo ne permet même pas le stockage d'une vidéo ayant une durée de 2 heures (à peine l'équivalent d'un film).
Les images vidéo ayant une taille minimum de 1 Mo (cela correspond à une vidéo de qualité inférieure), un disque dur de 120 Go peut contenir 122 880 images vidéo au maximum. Etant donné que la norme du cinéma, permettant de donner une animation fluide est de 24 images par seconde, ce disque dur ne peut contenir que 85 minutes de film vidéo. C'est pourquoi, afin de résoudre le problème d'encombrement, il est important de compresser l'information numérique.

A – Le texte numérique

Contrairement à l'homme, l'ordinateur n'est pas capable de comprendre un texte car il est limité au calcul sur des nombres. Il a donc fallu décider d'une représentation des caractères en termes de nombres compréhensibles pour l'ordinateur.
La convention la plus connue pour la représentation des caractères en binaire est le code ASCII[4], qui est l'un des standards les plus utilisés sur la plupart des ordinateurs. Ce standard code chaque caractère en une combinaison de 7 bits, c'est-à-dire en une valeur numérique comprise entre 0 et 127, ou encore en une valeur hexadécimale comprise entre 00 et 7F.
Comme l'illustre le tableau II.2, dans la base hexadécimale, chaque digit hexadécimal représente une combinaison de 4 bits. Ainsi, par exemple, la valeur hexadécimale 4B correspond à la combinaison binaire `0100 1011`, et donc à la valeur numérique 75. De

[4] Pour American Standard Code for Information Interchange

manière symétrique, la combinaison binaire `0110 1111` (i.e. le nombre 111) a pour valeur hexadécimale 6F.
Le tableau II.3, que l'on peut trouver notamment dans [CAL04], donne la correspondance entre les symboles d'écriture et les valeurs hexadécimales (et donc les valeurs numériques, indirectement). Les symboles sur fond jaune ne sont pas des caractères imprimables.
On construit la valeur décimale d'un caractère, en prenant le digit de la ligne dans laquelle il se trouve, suivi du digit de sa colonne. Par exemple, la valeur hexadécimale de la lettre "z" est 7A, ce qui correspond à la suite binaire `0111 1010`, et donc au nombre 122. De cette façon, le mot ASCII peut être représenté numériquement, par la suite hexadécimale 41 53 43 49 49, (c'est-à-dire, `01000001 01010011 01000011 01010001 01010001`).

Tableau II.2. Base hexadécimale.

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

Tableau II.3. Code ASCII US défini par la norme iso-646.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STH	ETH	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	CD2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	spc	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Comme nous pouvons le constater, avec le code ASCII, 7 bits suffisent à coder la plupart des caractères. En réalité, bien que ce standard suffise à coder les caractères nécessaires à la langue anglo-américaine, il ne permet pas de représenter les lettres accentuées, par exemple. C'est pourquoi ce code a été étendu, par la norme iso-8859, utilisant un 8ème bit (i.e. les codes 128 à 255) pour représenter ces caractères spéciaux. Toutefois, cette norme définit 15 versions différentes, pour satisfaire à tous les besoins des différents pays, ce qui ne facilite pas l'échange international de documents. Pour cette raison, a été créé l'UNICODE[5] qui utilise les codes 0 à 65535 (i.e. de 0000 à FFFF en base 16).

B – L'image numérique

L'image numérique est soit une image vectorielle, soit une image matricielle.
L'image vectorielle est utilisée, par exemple, dans les logiciels de dessins industriels. Elle se compose de différentes formes géométriques simples, décrites de manière mathématique (cf. [CRDP04]). C'est pourquoi elle ne prend pas beaucoup de place et peut être redimensionnée sans qu'il n'y ait de perte d'information.
Dans le cas d'une image obtenue à l'aide d'un appareil photo numérique, une caméra numérique ou un scanner, elle est composée d'un tableau de pixels à deux dimensions. D'où son nom d'image matricielle. Un pixel est donc représenté dans l'espace par ses coordonnées (x, y) et est associé à une valeur de luminance (Y) et deux valeurs de

[5] Ce point sortant du cadre du mémoire, veuillez vous référer à [UNICODE], pour davantage d'informations à ce sujet.

chrominance (C_r et C_b). Alors que la luminance est un composant d'intensité représentant des informations sur le blanc et le noir, C_r et C_b sont des composants de couleurs. Il est possible d'exprimer ces trois valeurs en termes des valeurs d'un autre espace de couleurs connu sous le nom de RGB. Dans cet espace, chaque pixel est associé à une valeur correspondant à la quantité des couleurs rouge, verte et bleue qu'il contient. Les expressions II.1 à II.3 montrent comment passer d'un modèle à l'autre en exprimant, respectivement Y, C_r et C_b en fonction des composantes R, G et B. Les normes de compression JPEG et MPEG, que nous verrons plus loin, se basent sur l'espace de couleurs YC_rC_b.

$$Y = 0{,}299*R + 1{,}587*G + 0{,}114*B \quad \textbf{[II.1]}$$

$$C_r = -0{,}299*R - 0{,}587*G + 0{,}886*B \quad \textbf{[II.2]}$$

$$C_b = 0{,}701*R - 0{,}587*G + 0{,}114*B \quad \textbf{[II.3]}$$

C – Le son numérique

Pour convertir un son en valeurs numériques, il faut relever des petits échantillons de son à des intervalles de temps réguliers. Pour restituer un son qui semble continu à l'oreille, ces intervalles doivent être de l'ordre du 100 000ème de seconde. On associe à chaque échantillon ainsi obtenu une valeur déterminant la mesure de la pression de l'air à ce moment. Il faut déterminer le nombre de valeurs qu'un échantillon peut prendre, ce qui correspond au nombre de bits nécessaires à les coder. Bien sûr, plus cette valeur est grande, meilleure est la qualité. Il faut également définir le nombre de voies (une voie pour un son monophonique, deux pour du stéréophonique, etc.).
Un son est donc représenté sous forme numérique à l'aide des paramètres suivants ([CCM]):

- La fréquence d'échantillonnage.
- Le nombre de bits d'un échantillon.
- Le nombre de voies.

D – La vidéo numérique

L'image animée étant une succession rapide d'images fixes, la vidéo numérique se compose d'images matricielles.

II.2.3 – La compression de l'information

La compression consiste à réduire le volume des données.
Dans le cas de l'audio, étant donné que certains sons peuvent ne pas être perçus selon les conditions, certaines informations sonores sont supprimées. Cette technique se base sur le fait que l'oreille humaine ne perçoit les sons qu'à partir d'un certain seuil d'audibilité, et sur le fait que les sons forts masquent les sons faibles.
Pour la compression des images, il existe deux types de méthodes de réduction, l'une n'entraînant aucune perte et l'autre bien. Dans le premier type de méthode, dite de compactage, l'application successive des fonctions servant à compacter et à décompacter les données fournit une image identique à l'originale. Le second type de méthode, la compression, entraîne quant à lui des pertes de qualité non perceptible à l'œil humain.

Le compactage consiste à éliminer la redondance des données. Cette technique se base sur le fait qu'une image se compose de surfaces de couleurs identiques. Donc, au lieu

de coder chaque point de l'image, il est plus intéressant de coder la couleur et sa répétition. L'inconvénient de cette méthode est son manque d'efficacité pour les images plus détaillées telles que les photographies. C'est pourquoi des techniques de compression ont été mises au point. Elles se basent sur le fait que l'œil humain ne perçoit pas tout. Il s'agit des normes de compression JPEG et MPEG.

A – La compression JPEG

La norme de référence JPEG (pour Joint Photographic Experts Group) est souvent utilisée pour la compression des images fixes. Elle est également utilisée pour les images animées, dans un contexte où chaque image doit être codée indépendamment des autres.

$$f(x,y) = \frac{2}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} c(u)\, c(v)\, F(u,v)\ cos\ \frac{\Pi(2x+1)u}{2N}\ cos\ \frac{\Pi(2y+1)v}{2N} \qquad \textbf{[II.4]}$$

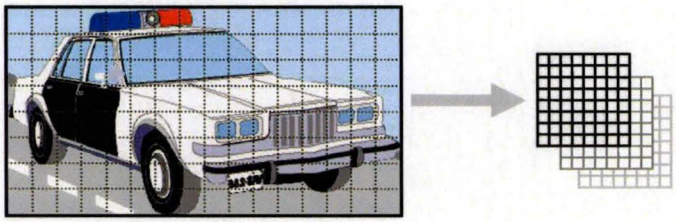
$$F(u,v) = \frac{2}{N} c(u)\, c(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y)\ cos\ \frac{\Pi(2x+1)u}{2N}\ cos\ \frac{\Pi(2y+1)v}{2N} \qquad \textbf{[II.5]}$$

avec $\begin{cases} c(0) = \frac{1}{\sqrt{2}} \\ c(i) = 1 \quad (i = 1, 2, ..., N-1) \end{cases}$

De manière générale, le principe de JPEG est d'exploiter la redondance contenue dans une image. Son fonctionnement est illustré à la figure II.3. Dans un premier temps, comme le montre la figure II.3 (a), l'algorithme de compression découpe l'image en blocs de 8x8 pixels, afin de faciliter les calculs. Ensuite, la méthode consiste à regarder, dans chaque bloc, le nombre de fois qu'un pixel donné apparaît. Cela se fait à l'aide d'une transformée en cosinus discrète (ou DCT[6]), dont un exemple est illustré à la figure II.3 (b). La DCT n'est rien d'autre que la dérivée de la transformée de Fourrier et est donnée à l'expression II.4 (prendre N=8), reprise de [KEZU02]. L'application de cette transformation sur un bloc[7], fournit un autre bloc de même dimension contenant non plus des pixels, mais 64 coefficients. Ces derniers représentent des écarts par rapport à la moyenne d'éclairement du bloc de départ. De cette façon on élimine la redondance de l'information dans chaque bloc.
Notons que dans la matrice transformée, les valeurs les plus élevées se retrouvent dans le coin supérieur gauche. Cela signifie que la majorité de l'information contenue dans l'image est concentrée dans les basses fréquences, ce qui facilitera la compression de l'image.

[6] Pour Discrete Cosine Transform.
[7] En réalité, selon [JPEG01], la DCT s'applique sur une matrice 16x16 obtenue par duplication du bloc, de manière symétrique, par rapport aux axes x et y.

(a)



(b)

140	144	147	140	140	155	179	175
144	152	140	147	140	148	167	179
152	155	136	167	163	162	152	172
168	145	156	160	152	155	136	160
162	148	156	148	140	136	147	162
147	167	140	155	155	140	136	162
136	156	123	167	162	144	140	147
148	155	136	155	152	147	147	136

DCT

186	-18	15	-9	23	-9	-14	-19
21	-34	26	-9	-11	11	14	7
-10	-24	-2	6	-18	3	-20	-1
-8	-5	14	-15	-8	-3	-3	8
-3	10	8	1	-11	18	18	15
4	-2	-18	8	8	-4	1	-7
9	1	-3	4	-1	-7	-1	-2
0	-8	-2	2	1	4	-6	0

(c)

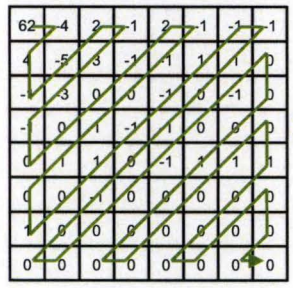
186	-18	15	-9	23	-9	-14	-19
21	-34	26	-9	-11	11	14	7
-10	-24	-2	6	-18	3	-20	-1
-8	-5	14	-15	-8	-3	-3	8
-3	10	8	1	-11	18	18	15
4	-2	-18	8	8	-4	1	-7
9	1	-3	4	-1	-7	-1	-2
0	-8	-2	2	1	4	-6	0

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31

quantification

62	-4	2	-1	2	-1	-1	-1
4	-5	3	-1	-1	1	1	0
-1	-3	0	0	-1	0	-1	0
-1	0	1	-1	-1	0	0	0
0	1	1	0	-1	1	1	1
0	0	-1	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(d)



62 -4 4 -1 -5 2 -1 3 -3 -1 0 0 0 -1 ...

(e)

... 3 -3 -1 0 0 0 -1 ... → ... 3 -3 -1 "3 0" -1 ...

(f)

Valeur	-1	1	2	62	4	-5	3	-3
Occurrence	13x	9x	2x	1x	1x	1x	1x	1x
Code	1	00	0110	0111	01000	01001	01010	01010

Figure II.3. Les étapes de codage JPEG.

Après cette étape de transformation, l'algorithme de compression réduit le nombre de bits nécessaires au stockage des valeurs du bloc. A cette fin, la matrice DCT va suivre une étape de quantification, comme nous le montre la figure II.3 (c). Lors de cette étape les valeurs $m_{i,j}$ qu'elle contient vont subir une division par les valeurs $q_{i,j}$ (ou quantum) d'une matrice de quantification. Il s'agit d'une matrice dont les valeurs sont d'autant plus élevées qu'elles s'éloignent de la position (0,0). De cette façon, les plus hautes fréquences de la matrice DCT seront le plus réduites, et vice versa. La matrice de quantification utilisée se construit sur base de l'expression II.6. Dans l'illustration, cette matrice, représentée ici en rouge, est construite avec un K valant 2. Etant donné que le résultat de la division n'est pas toujours une valeur entière, l'algorithme arrondit celui-ci. En conséquence, l'image codée au format JPEG contient des pertes par rapport à l'image originale.

$$q_{i,j} = 1 + K(1 + i + j) \qquad \textbf{[II.6]}$$

Après cette étape de quantification, l'algorithme va parcourir la matrice en zigzag, en allant du coin supérieur gauche au coin inférieur droit (cf. figure II.3 (d)). Les éléments parcourus de cette façon formeront une suite de valeurs contenant des grandes sous-suites de zéros. C'est cette suite de nombres qui sera enfin compressée, à l'aide d'une technique destinée aux zéros et une autre technique destinée aux valeurs non nulles. La première technique, dite de *codage à longueur variable*, consiste à remplacer les sous-suites de zéros par leur longueur (cf. figure II.3. (e)). Donc, plus il y a de suites de petite longueur, plus le code sera long. La seconde technique, appelée *codage entropique*, consiste à coder les éléments non nuls en fonction de leur occurrence. Comme nous pouvons le voir à la figure II.3 (f), plus une valeur se retrouvera dans la suite, plus son code sera court.

Le principe général pour le décodage d'une image JPEG consiste quant à lui à exécuter les étapes suivies pour le codage, en sens inverse. Ainsi, une image JPEG passera par des phases de décodages (correspondant aux deux techniques que nous venons de voir), de "déquantification", de transformation par DCT inverse (cf. expression II.5) avant de retrouver une forme proche à l'image originale.

Comme nous l'avons vu, l'inconvénient de JPEG est la perte de qualité au fur et à mesure des multiples compressions/décompressions. C'est pourquoi, dans certains contextes, il est impensable d'utiliser cette norme de compression.

B – La compression MPEG

Le groupe MPEG (pour Moving Picture Experts Group) travaille au développement de normes pour la compression des images animées et du son. La première norme développée par le groupe, MPEG-1, cible le stockage des contenus numériques. Ensuite vint MPEG-2, destinée à la télévision numérique, et enfin MPEG-4 qui s'applique à la diffusion de l'information numérique sur des canaux à bas débits.

MPEG-1

MPEG-1 se base sur le fait qu'une image animée est une succession rapide d'images fixes qui, souvent, ne varient que faiblement de l'une à l'autre. Donc, MPEG se base non seulement sur la redondance intra-images (comme le fait JPEG), mais également sur la redondance inter-images, c'est-à-dire sur le fait que deux images consécutives ne diffèrent que très peu.

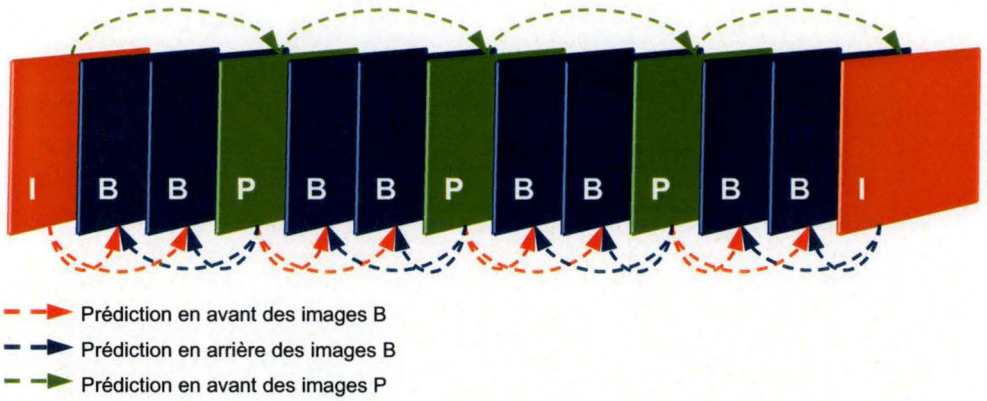


Figure II.4. Une séquence MPEG-1 pour M=3 et N=2.

De manière générale, comme le montre la figure II.4 (inspirée de [MPEG02]), la compression MPEG-1 consiste à enregistrer des images intervalles réguliers et à en déduire les autres images.

Les images I (*Intra Picture*), représentées en rouge dans l'illustration, sont intra-codées à l'aide de l'algorithme JPEG. Cela signifie qu'elles peuvent être reconstruites indépendamment des autres images. Les images P (*Predictive Picture*), représentées en vert, sont inter-codées puisqu'il s'agit de prédictions par rapport à la dernière image I ou P rencontrée. Donc, il est nécessaire de disposer de l'image de référence si l'on veut reconstruire l'image P. Les images B (*Bi-directionnaly Predictive Picture*), représentées en bleu, sont également inter-codées. Ce sont des prédictions par rapport à l'image I ou P précédente ainsi que l'image I ou P suivante. Ici aussi, les images de référence sont nécessaires à la reconstruction de l'image B.

Une séquence d'images est délimitée par deux images I. Elle est caractérisée par deux paramètres que sont la distance M séparant les deux images I et la distance N séparant deux images P. Plus les valeurs de ces paramètres sont grandes, meilleure est la qualité de codage. Par défaut, ces deux valeurs valent, respectivement 3 et 12, comme dans l'exemple illustré à la figure II.4.

Les prédictions dont il est question dans MPEG-1 sont, en réalité, des estimations de mouvement. Imaginons une image I sur laquelle on voit une branche de citronnier abritant 4 citrons. Une image P suivante montre la même branche et les mêmes citrons, mais l'un d'eux est tombé par terre. L'estimation de mouvement consiste à fournir un vecteur qui déclare comment, à partir de l'image I, il faut faire bouger le citron, pour obtenir l'image P. La figure II.5 (a) illustre ce propos. L'inconvénient est que cela suppose que chaque variation entre deux images s'exprime en un déplacement de pixels. Mais il se peut, par exemple, que le citron, en tombant, change de position. C'est-à-dire qu'en plus de subir une translation vers le sol, il subirait également une rotation de 35° vers la droite, comme le montre la figure II.5 (b). Dans ce cas, un simple déplacement du citron aurait pour conséquence une erreur de prédiction. C'est pourquoi MPEG-1 définit une matrice pour corriger cette erreur. De cette façon, la reconstruction des images inter-codées consiste, dans un premier temps, à appliquer le vecteur de mouvement sur l'image de référence, et à ajouter ensuite cette compensation à l'erreur de prédiction. En réalité, l'estimation de mouvement ne se fait pas sur l'image entière. En effet, l'image est divisée en macroblocs ayant une dimension de 16x16 pixels, contenant chacun 4 blocs de luminance et 2 blocs de chrominance. Chaque macrobloc a son propre vecteur de mouvement. Ainsi, il est possible d'estimer les mouvements de plusieurs éléments entre deux images de référence.

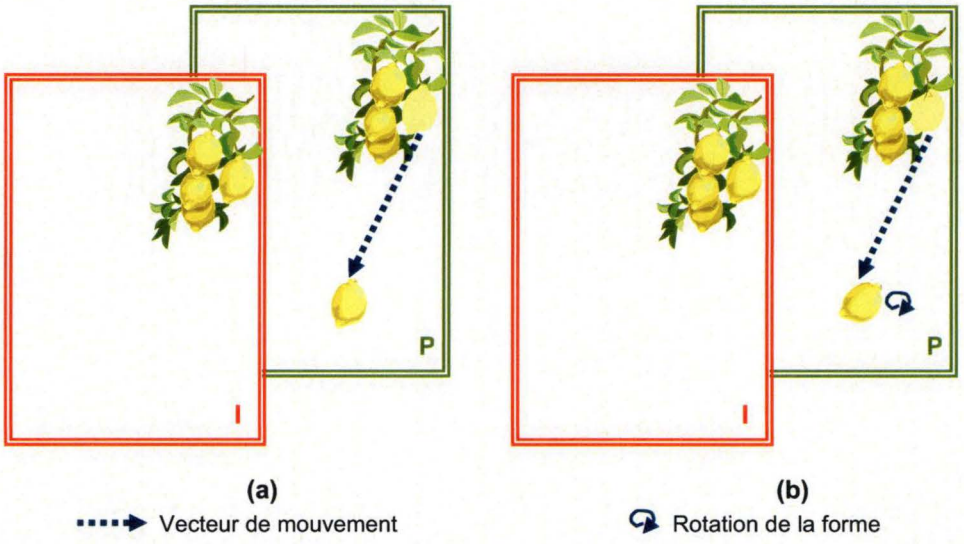


Figure II.5. Estimation du mouvement. **(a)** Translation d'un élément. **(b)** Translation et rotation d'un élément.

Il est à noter que la complexité de la norme se situe du côté de l'encodeur. Ceci permet de rendre le décodeur le plus simple possible. Cette complexité s'observe notamment dans l'ordre dans lequel la séquence MPEG est envoyée par l'encodeur. Celui-ci transmet tout d'abord l'image I de la séquence. Vient ensuite la première image P, suivie des images B se situant entre l'image I et cette image P. La suite de la séquence est transmise en inversant à chaque fois les images B et l'image P. De cette façon, la séquence illustrée à la figure II.4 est transmise par l'encodeur de la façon suivante: **IPBBPBBPBBIBB**. Cet ordre de séquence permet le décodage le plus efficace possible. Lors de la phase de décodage, la séquence est réordonnée de la même façon qu'illustrée à la figure II.4.

MPEG-2

MPEG-2 est une extension de MPEG-1 destinée au codage de vidéos pour la télédiffusion. Il tend à devenir un système de codage de vidéos générique supportant un grand nombre d'applications. Le codage d'une séquence vidéo MPEG-2 se base sur les techniques exploitées par MPEG-1. La particularité de la norme vient de la syntaxe des flux de bits vidéo. Certains éléments de cette syntaxe sont optionnels et des flags, situés dans d'autres éléments, signalent leur présence ou non. Cette "technique" rend la syntaxe variable, ce qui permet au système de codage d'être générique. Cela permet également de diminuer la quantité de données transmises (en ne transmettant pas les éléments optionnels lorsqu'ils ont une valeur nulle, par exemple).
Comme nous pouvons le voir à la figure II.6, issue de [BRTI02], la syntaxe vidéo est organisée en hiérarchie.

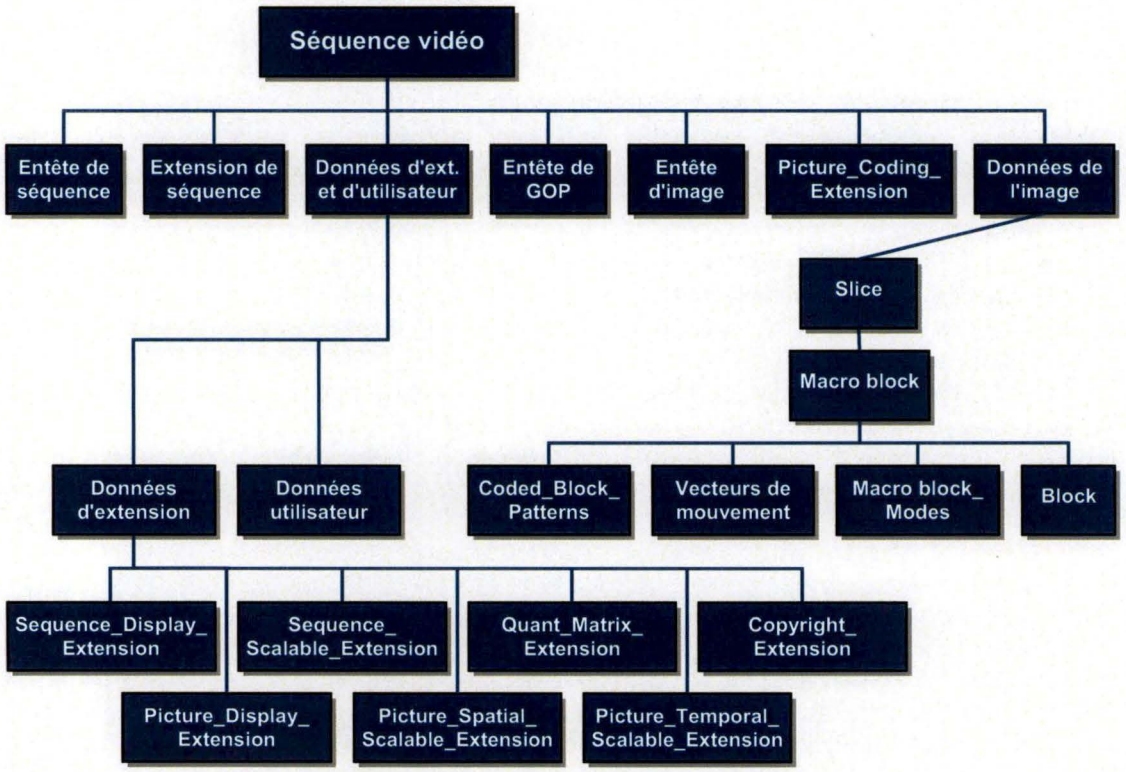


Figure II.6. Syntaxe vidéo organisée en hiérarchie.

Une séquence vidéo contient des images et des données d'extension. Elle est composée des éléments suivants:

- L'**Entête de séquence** contient des informations telles que la taille des images et le nombre d'images par seconde.
- L'**Extension de séquence** contient le profil, le niveau et le format de chrominance du flux de bits.
- Les **Données d'extension et d'utilisateur** regroupent des données d'extension et des données utilisateur.
- L'**Entête GOP** (Group Of Pictures) contient un timecode et des informations sur les images du groupe.
- L'**Entête de l'image** permet de connaître le type de l'image (i.e. I, P ou B).
- Le **Picture_Coding_Extension** contient des informations sur les images afin de supporter le balayage entrelacé ou le balayage progressif et les standards vidéo analogiques tels que NTSC ou PAL.
- Les **Données de l'image** contiennent des tranches (slices).
- Les **Slices** (ou tranches) contiennent la position verticale de la tranche, l'information concernant le partitionnement des données, l'échelle de quantification et sont composés de macroblocs.
- Les **Macro blocks** contiennent une échelle de quantification (optionnelle) et comprennent les Coded_Block_Patterns, les vecteurs de mouvement, les Macro block_Modes et toute une série de blocs.
- Les **Coded_Block_Patterns** indiquent les blocs du macrobloc qui sont réellement encodés.
- Les **Vecteurs de mouvement** contiennent les vecteurs de mouvement pour un macrobloc donné.
- Les **Macro block_modes** indiquent le type d'encodage utilisé pour le macrobloc.

- Le **Block** contient des coefficients DCT.
- Les **Données utilisateur** concerne des données définies par l'utilisateur.
- Les **Données d'extension** concernent toute une série d'extensions.
- La **Sequence_Display_Extension** contient des informations supplémentaires pour l'affichage de la séquence (le format vidéo et les attributs de couleur utilisés dans le flux de bits).
- La **Sequence_Scalable_Extension** contient des informations supplémentaires pour l'évolutivité.
- La **Quant_Matrix_Extension** contient des matrices pour la quantification inverse.
- La **Copyright_Extension** indique si on a affaire à un flux original ou à une copie.
- La **Picture_Display_Extension** contient des informations supplémentaires pour l'affichage de l'image.
- La **Picture_Temporal_Scalable_Extension** contient des informations pour l'évolutivité temporelle.
- La **Picture_Spatial_Scalable_Extension** contient des informations pour l'évolutivité spatiale.

Certaines applications peuvent ne pas avoir besoin de toutes les caractéristiques de la norme. C'est pourquoi MPEG-2 définit des sous-ensembles organisés en 5 profils et en 4 niveaux. Les tableaux II.4 et II.5 (issus de [BRTI02]) reprennent respectivement les caractéristiques MPEG-2 selon les profils et selon les niveaux. Les profils définissent le format de chrominance, le type d'images et le mode de *scalabilité* utilisables. Les niveaux définissent des valeurs pour certains paramètres dans le flux des bits vidéo.

Tableau II.4. Les caractéristiques MPEG-2 en fonction des profils.

Caractéristiques MPEG-2	Profil				
	Simple	Principal	SNR[8]	Spatial	Elevé
Format de chrominance	4:2:0	4:2:0	4:2:0	4:2:0	4:2:0 ou 4:2:2
Type d'image	I, P	I, P, B	I, P, B	I, P, B	I, P, B
Mode de *scalabilité*	-	-	SNR	SNR ou spatial	SNR ou spatial

Tableau II.5. Les caractéristiques MPEG-2 en fonction des niveaux.

Caractéristiques MPEG-2	Niveau			
	Bas	Principal	Haut 1440	Haut
Résolution	325*240*30	704*480*30	1440*1152*30	1920*1080*30
Pixel/seconde (millions)	3,05	10,4	40	62,7
Débit maximum (Mb/s)	4	15	60	80
Taille de mémoire tampon (bits)	475 136	1 835 008	7 340 032	9 781 248

Il y a donc 20 combinaisons possibles de profils et de niveaux, ce qui correspond à autant de sous-ensembles de caractéristiques vidéo. Toutefois MPEG-2, comme nous pouvons le voir dans le tableau II.6, MPEG-2 ne reconnaît que 11 combinaisons.

[8] Signal-to-Noise Ratio.

Tableau II.6. Les combinaisons de profils et niveaux reconnues par MPEG-2.

		Profil				
		Simple	Principal	SNR	Spatial	Elevé
Niveau	Bas		☑	☑		
	Principal	☑	☑	☑		☑
	Haut 1440		☑		☑	☑
	Haut		☑			☑

MPEG-4

MPEG-4 se destine à des débits très faibles pour permettre des applications interactives (comme, par exemple, des vidéoconférences) transitant dans des réseaux téléphoniques.
Une scène audiovisuelle MPEG-4 est composée de plusieurs objets média organisés en hiérarchie. Les feuilles de la hiérarchie peuvent être des images fixes (par exemple, un décor), des objets vidéo (par exemple, un conférencier, sans le décor) ou des objets audio (par exemple, la voix de cette personne). En outre, MPEG-4 définit la représentation codée d'objets tels que:

- Du texte et des graphiques.
- Des têtes parlantes synthétiques et le texte associé, pour synthétiser le discours et animer la tête; des corps animés qui vont avec la tête.
- Du son synthétique.

Lors du codage, la scène est décomposée en objets qui sont codés et transmis séparément. Le décodeur aura alors pour but de reconstruire l'image à partir des différents objets et des informations de composition de l'image.

La figure II.7, issue de [ISO02a], illustre la façon dont une scène MPEG-4 est décrite en tant que composition d'objets individuels. La figure contient des objets média composés qui regroupent plusieurs objets médias primitifs. Comme nous venons de le voir, les objets médias primitifs correspondent aux feuilles de la hiérarchie. Les objets média composés, quant à eux comprennent des sous-arbres entiers. Par exemple, l'objet visuel correspondant au conférencier et la voix associée sont reliés ensemble pour former un nouvel objet média composé, contenant les deux composants audio et visuel du conférencier.
Grâce à de tels groupements, les auteurs peuvent reconstruire des scènes complexes et les consommateurs peuvent manipuler des ensembles d'objets significatifs.

Donc, de manière générale, la norme fournit un moyen standard pour décrire une scène audiovisuelle, en permettant notamment:

- De placer les objets média n'importe où dans un système de coordonnées donné.
- D'appliquer des transformations pour changer l'apparence géométrique ou acoustique d'un objet média.
- De grouper des objets média primitifs de façon à former des objets média composés.
- De changer les points de vision et d'audition de l'utilisateur, n'importe où dans la scène.

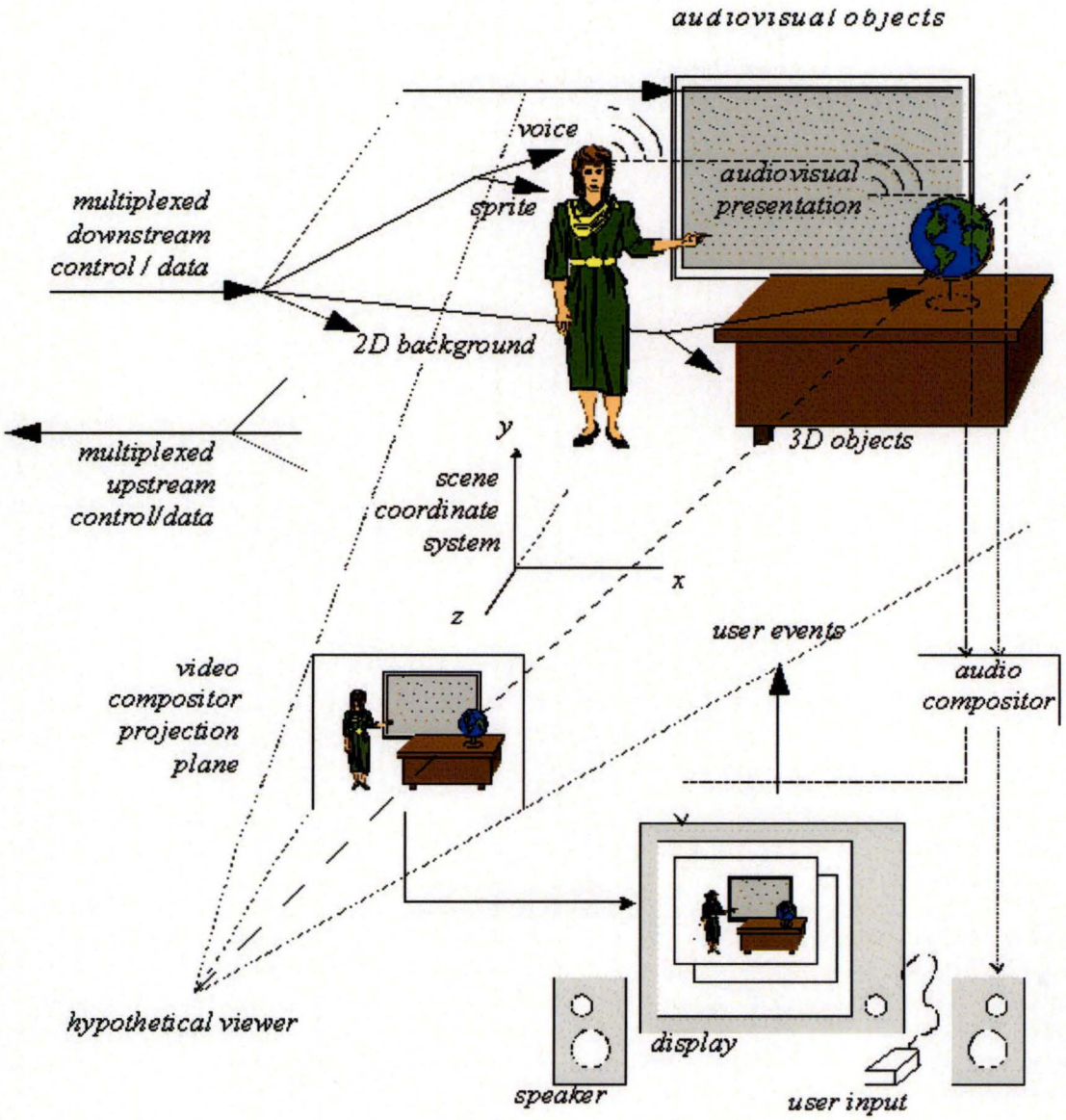


Figure II.7. Un exemple de scène MPEG-4.

Le groupe MPEG est également à l'origine du développement de la norme MPEG-7 ayant pour cible la description des documents multimédia. Nous aborderons ce point dans la section II.3.

Enfin, le dernier enfant en date du groupe MPEG est un standard pour la consommation et la diffusion du multimédia, appelé MPEG-21[9]. Il s'agit d'un cadre (*framework*) ouvert destiné à tous les acteurs de la chaîne de diffusion et de consommation (créateurs de contenu, producteurs, distributeurs, fournisseurs de services et consommateurs de contenu).
MPEG-21 se base sur deux concepts: la définition d'une unité de distribution et de transaction de base, appelée **Objet Numérique** (*Digital Item*) et le concept d'**Utilisateurs** (*Users*) interagissant avec ceux-ci. Les Objets Numériques sont considérés comme le "quoi"

[9] Nous nous limiterons ici à introduire les concepts-clés de MPEG-21. Le lecteur intéressé pourra trouver davantage d'informations dans [ISO02b].

du framework multimédia, alors que les Utilisateurs sont considérés comme le "qui" du framework multimédia.
Un exemple d'Objet Numérique serait, par exemple, la présentation d'une université, incluant des photos, des vidéos, des graphiques animés, de l'information textuelle, des nouvelles liées aux activités de recherche, du matériel d'apprentissage en ligne, de l'information concernant la navigation en fonction des préférences des utilisateurs, etc. Cet exemple est illustré à la figure II.8, issue de [BUR+03].

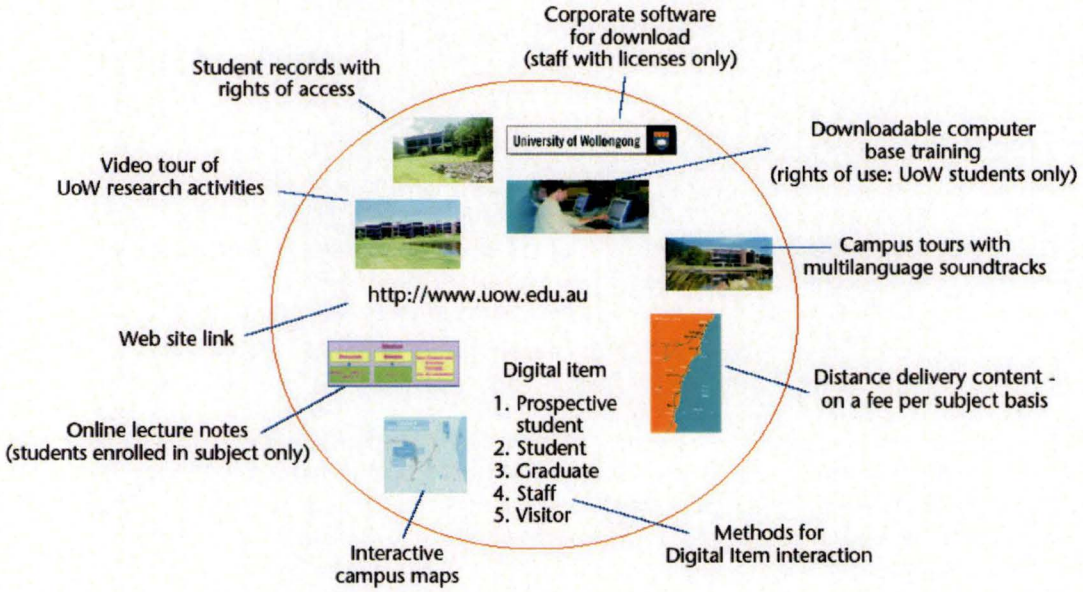


Figure II.8. Exemple d'Objet Numérique MPEG-21.

Le but de la norme peut donc se résumer à la définition de la technologie nécessaire pour aider les Utilisateurs à échanger, accéder, consommer et manipuler les Objets Numériques de manière efficace et transparente.

De nos jours, une information audiovisuelle pouvant prendre différentes formes et provenir de partout dans le monde est de plus en plus présente et disponible.
Bien que, dans la plupart des cas, nous nous limitions à consommer directement l'information audio et visuelle, de plus en plus de systèmes informatiques créent, échangent, recherchent et réutilisent cette information. Pensons, par exemple, aux cas de reconnaissance d'images ou de conversion de média. N'oublions pas, non plus, les systèmes de récupération d'information où il est nécessaire de rechercher rapidement et de manière efficace différents types de documents pertinents pour l'utilisateur.

De plus en plus, les informations multimédia vont jouer un rôle important dans notre vie, et il y a une nécessité croissante de traiter davantage ces ressources. Jusqu'à présent, les schémas de représentation (PAL, NTSC[10], MPEG-1, MPEG-2, MPEG-4, etc.) ont évolués pour fournir les meilleurs moyens de transmission, de stockage, et de recherche de l'information multimédia, pour la consommation humaine. Mais aujourd'hui, nous avons besoin de formes de représentation permettant un certain degré d'interprétation du sens des définitions auxquelles peuvent accéder un dispositif ou un code informatique. Et c'est précisément ce que supporte le standard MPEG-7, qui fait l'objet de la prochaine section.

[10] PAL et NTSC sont tous deux des normes de codage des couleurs pour la diffusion et l'enregistrement des images vidéo.

II.3 – MPEG-7: Un standard de description pour les contenus multimédia

Le "Multimedia Content Description Interface" (MPEG-7) est un standard ISO de métadonnées visant à fournir des outils pour la description de contenus multimédia utilisée dans de nombreuses applications ([MAR02a] et [MAR02b]).
MPEG-7 ne vise pas une application en particulier; mais, au contraire, le plus d'applications possible.

Il faut considérer un grand nombre de fonctions descriptives du contenu multimédia pour couvrir toutes ces applications. Mais un sous-ensemble seulement de ces fonctions devra être utilisé, selon le contexte de l'application visée. Cela signifie qu'il n'existe pas une seule description correcte pour un contenu donné; il existe plusieurs descriptions différentes possibles dont toutes sont équivalentes.

MPEG-7 se distingue des autres standards de métadonnées dans le fait qu'il supporte un grand nombre de niveaux d'abstraction, allant des caractéristiques de signal de bas niveau à l'information sémantique de haut niveau.
Le niveau d'abstraction concerne la manière dont les caractéristiques sont extraites. Alors qu'il est possible d'extraire de façon automatique la plupart des caractéristiques de bas niveau, une intervention humaine est nécessaire pour superviser et annoter les caractéristiques de haut niveau. MPEG-7 fixe uniquement le format de description, et non les méthodes d'extraction.

MPEG-7 doit également supporter la description d'autres types d'information en ce qui concerne les données multimédia telles que la forme, le lieu et l'heure d'enregistrement, la classification et les liens vers les autres matériaux pertinents.
Dans ce standard, les descriptions sont d'abord générées, en utilisant le format MPEG-7. Elles sont ensuite stockées ou écoulées pour être enfin consommées en fonction de l'application en question.

La production de la description commence par une extraction des caractéristiques (au moyen d'une analyse de contenu) ou une annotation. L'extraction automatique des caractéristiques pour exploiter pleinement les possibilités des descriptions MPEG-7 serait extrêmement utile. Toutefois, ce n'est pas toujours possible: au plus le niveau d'abstraction est grand, au plus l'extraction automatique est difficile et, par conséquent, l'utilisation d'outils interactifs sera requise (pour la supervision et l'annotation par des humains).

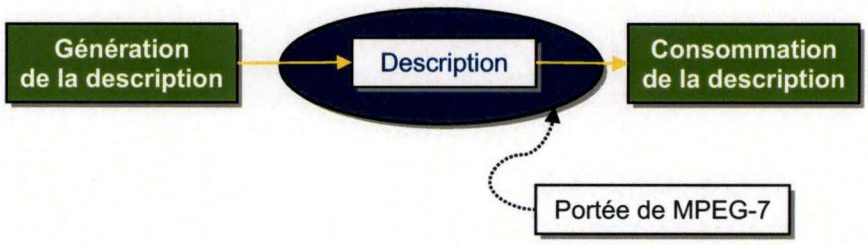


Figure II.9. Portée de MPEG-7.

Mais, aussi utiles qu'ils puissent être, comme nous l'illustre la figure II.9 ([ISO03]), on ne trouve aucun algorithme d'extraction automatique, ni semi-automatique dans la portée de

la norme. Ceci est dû au fait que l'interopérabilité ne requiert pas leur standardisation. Un autre argument à cette *limite* est que cela appelle à la compétition entre les différents fournisseurs et donc, à l'innovation.

Le standard MPEG-7 ne spécifie pas non plus la consommation au bout de la chaîne, c'est-à-dire la conception de n'importe quel programme qui peut faire usage de description, comme les moteurs de recherche, par exemple. Ici encore, ce n'est pas nécessaire ou la compétition et l'innovation donneront de meilleurs résultats. De plus, il est impossible de prédire les types d'applications qui utiliseront le format de Description MPEG-7. Cela signifie que la standardisation concerne uniquement le format de Description.

MPEG-7 standardise essentiellement quatre éléments que sont les Descripteurs (*Descriptors*), les Schémas de Description (*Description Schemes*), le Langage de Définition des Descriptions ou DDL (*Description Definition Language*) et les Outils Système (*System Tools*):

- Les **Descripteurs** (D) définissent la syntaxe et la sémantique de chaque caractéristique. Un Descripteur est donc la représentation d'une caractéristique.
- Les **Schémas de Description** (DS) spécifient la structure et la sémantique des relations entre leurs composants. Ceux-ci sont soit des Descripteurs, soit des Schémas de Description.
- Le **DDL** permet de créer de nouveaux Schémas de Description, et éventuellement des Descripteurs. Il permet également d'étendre et de modifier des Schémas de Description existant.
- Les **Outils Systèmes** permettent le support d'une représentation binaire pour un stockage et une transmission efficaces. Ils supportent également des mécanismes de transmission (au format binaire ou textuel), le multiplexage des descriptions, la synchronisation des descriptions avec le contenu, ainsi que la gestion et protection de la propriété intellectuelle dans les descriptions MPEG-7.

Ces définitions, données par [ISO01], se basent sur une terminologie selon laquelle une **donnée** (Data) est *"une information audiovisuelle qui sera décrite à l'aide de MPEG-7, quelque soit le stockage, le codage, l'affichage, la transmission, le support, ou la technologie".* De plus, on entend par **caractéristique** (*Feature*) *"une caractéristique distinctive de la donnée qui indique quelque chose à quelqu'un".*

Ces outils permettent la création et le déploiement de différents types de Descriptions (*Descriptions*).
Conformément à [ISO01], une **Description** consiste en un Schéma de Description (structure) et un ensemble de Valeurs de Descripteur (instanciations) qui décrivent la donnée. Une **Valeur de Descripteur** (*Descriptor Value*) est l'instanciation d'un Descripteur pour un certain ensemble (ou sous-ensemble) de données.

Les Descriptions classiques, orientées archivage comportent ([MAR02a]):

- L'information concernant la création de contenu et les procédés de production (directeur, titre, acteurs, lieu, etc.).
- L'information concernant l'utilisation du contenu (horaire de diffusion et pointeurs de copyright).
- L'information sur le rangement et la représentation du contenu (stockage et format d'encodage).

Mais il existe d'autres types de Descriptions qui comprennent, par exemple, de l'information spatio-temporelle, de l'information sur les caractéristiques de bas niveau (par exemple, la

couleur, le son ou les textures) et de l'information sur la réalité capturée par le contenu (par exemple, les objets et leurs relations entre eux ou les événements).
Un autre type d'information que peut comporter une Description est celle qui concerne l'organisation, la gestion et l'accès au contenu (la façon dont les objets sont reliés et groupés en collections, l'information permettant le support de parcours efficace du contenu et l'information concernant l'interaction de l'utilisateur avec le contenu).
Tous ces types de Descriptions sont corrélés et peuvent être combinés dans un Schéma de Description unique.

La figure II.10, issue de [ISO03], illustre la relation entre les différents éléments de MPEG-7. Le DDL permet de définir les outils de description MPEG-7 (Descripteurs ou Schémas de Description), fournissant les moyens de structurer le Schéma de Description en Schémas de Description. Le DDL permet également l'extension des Schémas de Description pour des applications spécifiques. Les outils de description sont instanciés comme descriptions au format XML à l'aide du DDL. Le format binaire des descriptions est obtenu au moyen du BiM défini dans [ISO03].

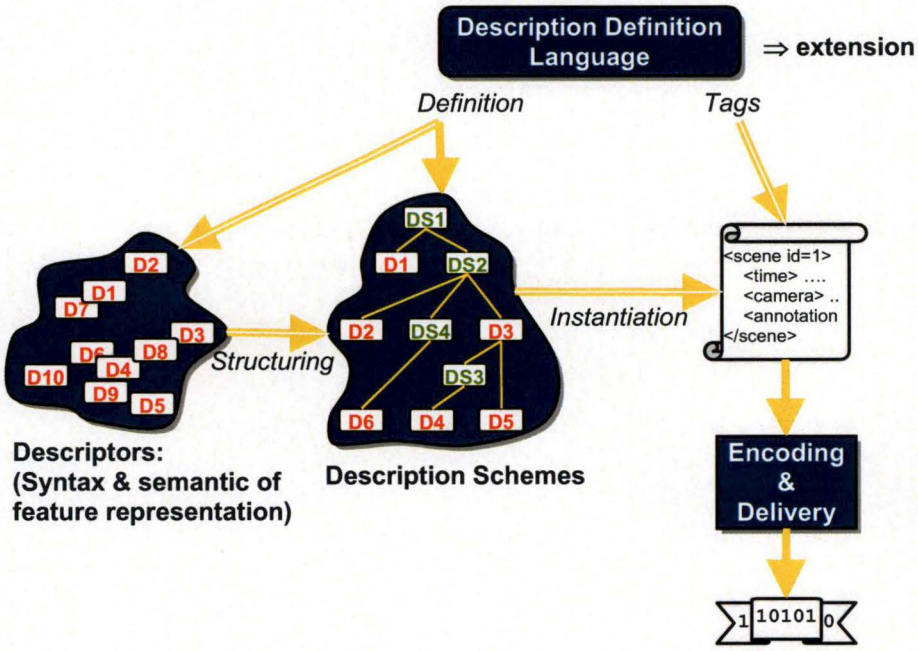


Figure II.10. Les éléments principaux de MPEG-7.

MPEG-7 tend à fournir de nouvelles solutions pour décrire le contenu multimédia. C'est pourquoi, afin de rencontrer les objectifs du standard, il ne faut pas s'adresser qu'à des documents textuels. Mais, il n'est pas rare que le contenu multimédia inclue ou se réfère à du texte, en plus de l'information audiovisuelle. C'est pourquoi MPEG-7 contient différents outils de descriptions pour les annotations, par exemple.

Les principales fonctionnalités[11] de MPEG-7 sont:

- Le **DDL** (*MPEG-7 DDL*), comme nous l'avons vu, permet la création de nouveaux Schémas de Description et Descripteurs, ainsi que l'extension et la modification de Schémas de Description existants.

[11] La norme comprend d'autres fonctionnalités comme *MPEG-7 Systems*, *MPEG-7 Conformance* ou encore *MPEG-7 Extraction*, mais celles-ci sortent du cadre de ce mémoire. Veuillez vous référer à [ISO03] et [BRTI02] pour davantage d'informations à ce sujet.

- Le **Visuel** (*MPEG-7 Visual*) fournit des outils constituant d'une par les structures de base et d'autre part les Descripteurs couvrant les caractéristiques visuelles de bases telles que la couleur, la texture, la forme, etc.).
- L'**Audio** (*MPEG-7 Audio*) réunit les structures permettant de décrire le contenu audio. Il s'agit de Descripteurs de bas niveau pour les caractéristiques traversant un grand champ d'applications (par exemple, les caractéristiques spectrales ou temporelles d'un signal) ainsi que de haut niveau, plus spécifiques à un ensemble d'applications.
- Les **Schémas de Description Multimédia** ou MDS (*MPEG-7 Multimedia Description Schemes*) regroupent un ensemble d'outils concernant aussi bien des entités génériques que des entités multimédia.

On entend par entité générique, les caractéristiques communes aux descriptions audio et visuelle, et donc à tous les média (i.e. les outils de description textuelle, le temps, etc.).

II.3.1 – Le DDL

MPEG-7 est extrêmement lié à XML, le langage de définition de schéma défini par le W3C, puisque le DDL est un *super-ensemble* du schéma XML.
Etant donné que le schéma XML n'est pas spécifiquement destiné au contenu multimédia, il était nécessaire d'ajouter certaines spécificités MPEG-7. C'est ainsi que le DDL comprend les éléments suivants:

- Des composants Structurels du Schéma XML ([XMLa]).
- Des composants concernant les Types de données du Schéma XML ([XMLb]).
- Des extensions MPEG-7 au schéma XML.

A – Les structures et les types de données du schéma XML

Les *Structures* ont pour rôle de décrire tant la structure que le contenu des documents XML 1.0. Les composants structurels d'un Schéma XML peuvent se décomposer en trois groupes. Ce sont les composants primaires, les composants secondaires et les composants d'aide qui contribuent aux deux précédents et qui donc n'ont pas lieu d'exister seuls.

Le premier groupe comprend:

- Le Schéma lui-même, c'est-à-dire l'emballage qui se trouve autour des définitions et des déclarations.
- Les définitions de types simples (i.e. des types de données qui n'ont pas d'élément fils ni d'attribut).
- Les définitions de types complexes (i.e. des types qui peuvent avoir plusieurs attributs ainsi que des éléments fils, ou qui peuvent être dérivés d'autres types simples ou complexes).
- Les déclarations d'attributs.
- Les déclarations d'éléments.

Il est à noter que les définitions de type définissent des composants de schéma internes pouvant être utilisés dans les déclarations d'éléments ou d'attributs, ou encore dans d'autres définitions de type.

Le deuxième groupe se compose:

- Des définitions de groupe d'attributs.
- Des définitions de contraintes d'intégrité.
- Des définitions de groupes nommés.
- Des déclarations de notation.

Les *Types de données*, quant à elles, permettent de définir des types de données à utiliser dans les Schémas XML en fournissant

- Des types de données primitives.
- Des types de données dérivées.
- Des mécanismes pour aider l'utilisateur à définir ses propres types de données dérivées.

A titre d'illustration, la figure II.11 fournit un exemple de schéma XML pour la gestion des horaires d'examen, ainsi que des résultats. Dans cette illustration, la ligne suivante permet d'associer l'espace de nom de XML Schema avec le préfixe `xsd`:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
targetNamespace="http://www.w3.org/2000/10/XMLSchema" version="1.1">
```

Cela se fait à l'aide de l'attribut `xmlns`. Donc, pour être valide, un élément appartenant au vocabulaire de XML Schema doit suivre directement `xsd:`. L'attribut targetNamespace, quant à lui permet de préciser un espace de nom par défaut, c'est-à-dire un espace de nom à utiliser lorsqu'un élément n'est pas préfixé. Dans ce cas ci, l'espace de nom par défaut (implicite) est identique à l'espace de nom explicite, mais il se peut que cela ne soit pas le cas.

Ensuite vient la déclaration des éléments:

```
<xsd:element name="remarque" type="xsd:string"/>
<xsd:element name="etudiant" type="typeEtudiant"/>
<xsd:element name="horaire" type="typeHoraire"/>
<xsd:element name="resultat" type="typeResultat"/>
```

Le schéma déclare 4 éléments que sont *remarque*, *etudiant*, *horaire* et *resultat*. A chaque élément est associé un type de données via l'attribut `type`.
Ainsi, le type de l'élément *remarque* (i.e. `xsd:string`) est un type simple, prédéfini de XML Schema. Les éléments de cette catégorie ne contiennent pas de sous-éléments. En revanche, les éléments *etudiant*, *horaire* et *resultat* ont pour types *typeEtudiant*, *typeHoraire* et *typeResultat*, respectivement. Il s'agit de types complexes, qui sont définis par l'utilisateur. Les éléments de type complexe, eux, peuvent contenir des sous-élément et peuvent porter des attributs.
Un attribut est un qualificatif qui s'applique à un élément. Un attribut ne peut être que de type simple et doit être déclaré à la fin des définitions de types complexes. Il en va de même pour les références aux groupes d'attributs.

Le fragment suivant est la déclaration du type simple *numeroDeTelephone*:

```
<simpleType name="numeroDeTelephone">
   <listitemType="xsd:unsignedByte"/>
</simpleType>
```

Les types listes sont des suites de types atomiques. Dans ce cas-ci, un numéro de téléphone sera représenté par une liste d'octets non signés. Donc, pour être conforme au

type *numeroDeTelephone*, un élément telephone (en supposant que celui-ci soit de type *numeroDeTelephone*) doit avoir l'allure suivante:

```
<telephone>02 24 15 74 48</telephone>
```

```
< ?xml version= "1.0" encoding="ISO-8859-1">

<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
targetNamespace="http://www.w3.org/2000/10/XMLSchema" version="1.1">

    <xsd:element name="remarque" type="xsd:string"/>
    <xsd:element name="etudiant" type="typeEtudiant"/>
    <xsd:element name="horaire" type="typeHoraire"/>
    <xsd:element name="resultat" type="typeResultat"/>

    <simpleType name="numeroDeTelephone">
       <listitemType="xsd:unsignedByte"/>
    </simpleType>

    <simpleType name='typeResultat">
       <unionmemberTypes="xsd:string typeCote"/>
    <simpleType>

    <complexType name="typeCote">
       <simpleContent>
          <extension base="xsd:positiveInteger">
             <attribute name="decision" type="xsd:string"/>
          </extension>
       </simpleContent>
    </complexType>

    <complexType name="typeEtudiant">
       <sequence>
          <element name="nom" type="xsd:string"/>
          <element name="prenom" type=" xsd:string"/>
          <element name="numeroEtudiant" type="xsd:positiveInteger"/>
          <element name="dateDeNaissance" type="xsd:date"/>
          <choice>
             <element name="adresse" type="xsd:string"/>
             <element name="adresseKot" type="xsd:string"/>
          </choice>
          <element name="email" type="xsd:string"/>
          <element name="telephone" type="xsd:numeroDeTelephone" minOccurs="0"/>
       </sequence>
    </complexType>

    <complexType name="typeHoraire">
       <element name="examen" type="xsd:string"/>
       <element name="professeur" type="xsd:string"/>
       <attributeGroup ref="InfosExam"/>
    </complexType>

    <attributeGroup name="InfosExam"/>
       <attribute name="date" type="xsd:date"/>
       <attribute name="local" type="xsd:string"/>
       <attribute name="mode" type="xsd:string"/>
    </attributeGroup>

</xsd:schema>
```

Figure II.11. Un exemple de Schéma XML pour la gestion d'horaires et de résultats d'examens.

XML Schema fournit également un autre type simple avec l'union. La déclaration du type simple *typeResultat* illustre cette notion:

```
<simpleType name='typeResultat">
   <unionmemberTypes="xsd:string typeCote"/>
<simpleType>
```

Grâce à l'union, un élément ou un attribut peut avoir comme type une réunion de plusieurs types atomiques ou listes. Dans ce cas-ci, le type *TypeResultat* est soit un string, soit le type *typeCote* dont la déclaration est réalisée par le fragment suivant:

```
<complexType name="typeCote">
   <simpleContent>
      <extension base="xsd:positiveInteger">
            <attribute name="decision" type="xsd:string"/>
      </extension>
   </simpleContent>
</complexType>
```

Le type *typeCote* est un type complexe créé à partir du type simple `positiveInteger`. Imaginons que l'on désire associer à une cote d'examen la décision de réussite (réussite, échec, report) de sorte à ce qu'un élément de type *typeCote* ait l'allure suivante:

```
<cote decision=echec>6</cote>
```

En supposant que l'élément *cote* soit de type *typeCote*.
Etant donné qu'un élément de type simple ne peut pas contenir d'attribut, il est impossible de déclarer l'élément cote comme étant de type `positiveInteger`. C'est pourquoi le type complexe *typeCote* est dérivé du type simple `positiveInteger`. Donc, en supposant que l'élément *resultat* soit de type *typeResultat*, les éléments suivants sont valides:

```
<resultat>satisfaction</resultat>
<resultat decision="reussite">12</resultat>
```

La déclaration suivante est celle du type complexe *typeEtudiant*:

```
<complexType name="typeEtudiant">
   <sequence>
      <element name="nom" type="xsd:string"/>
      <element name="prenom" type=" xsd:string"/>
      <element name="numeroEtudiant" type="xsd:positiveInteger"/>
      <element name="dateDeNaissance" type="xsd:date"/>
      <choice>
         <element name="adresse" type="xsd:string"/>
         <element name="adresseKot" type="xsd:string"/>
      </choice>
      <element name="email" type="xsd:string"/>
      <element name="telephone" type="xsd:numeroDeTelephone"
        minOccurs="0"/>
   </sequence>
</complexType>
```

Le connecteur de séquence, `sequence`, sert à définir un type complexe contenant des suites d'éléments (i.e. les éléments fils de l'élément `sequence`). Ainsi, dans l'exemple, le type *typeEtudiant* est composé d'un nom, d'un prénom, d'un numéro d'étudiant, d'une date de naissance, d'une adresse, d'une adresse e-mail et d'un numéro de téléphone. Les instances d'éléments de type *typeEtudiant* devront impérativement contenir ces éléments dans le même ordre que la déclaration.
Le connecteur de choix, `choice`, permet qu'une instance d'un élément de type *typeEtudiant* contienne soit l'adresse de domiciliation, soit l'adresse du kot de l'étudiant, mais en aucun cas les deux.

Il est possible qu'un élément d'un schéma comporte les attributs `minOccurs` ou `maxOccurs`. Cela correspond, respectivement, au nombre minimum et maximum d'occurrence de l'élément. Pour fixer cette valeur, on peut utiliser n'importe quel entier non négatif. Ainsi, dans l'exemple, le numéro de téléphone de l'étudiant est optionnel puisqu'il doit survenir au minimum 0 fois.

Si l'on suppose que etudiant est de type *typeEtudiant*, l'élément suivant est donc valide:

```
<etudiant>
    <nom>Robinet</nom>
    <prenom>Aude</prenom>
    <numeroEtudiant>19990423</numeroEtudiant>
    <dateDeNaissance>1981-08-25</dateDeNaissance>
    <adresse>10 Rue des Tournesols 5000 Namur BELGIQUE</adresse>
    <email>arobinet@info.fundp.ac.be</email>
</etudiant>
```

Le fragment suivant déclare le type complexe *typeHoraire*:

```
<complexType name="typeHoraire">
    <element name="examen" type="xsd:string"/>
    <element name="professeur" type="xsd:string"/>
    <attributeGroup ref="InfosExam"/>
</complexType>
```

typeHoraire est composé des éléments *examen* et *professeur*, qui sont tous les deux de type simple. Il contient également une série d'attributs que définit le groupe d'attribut *infosExam*:

```
<attributeGroup name="InfosExam"/>
    <attribute name="date" type="xsd:date"/>
    <attribute name="local" type="xsd:string"/>
    <attribute name="mode" type="xsd:string"/>
</attributeGroup>
```

Le type complexe *typeHoraire* possède donc les attributs *date*, *local* et *mode*, tous les trois de type `string`. L'avantage du groupe d'attribut est qu'il peut être défini à un seul endroit et référencé dans beaucoup de déclarations et de définitions. Il permet donc de rendre le schéma plus lisible et d'en faciliter la maintenance.

Le fragment suivant donne une instance d'élément valide de l'élément horaire, pour autant qu'il soit de type *typeHoraire*:

```
<horaire date=2004-06-20 local="I2" mode="ecrit">
      <examen>recherche operationnelle</examen>
      <professeur>J. Fichefet</professeur>
</horaire>
```

B – Les extensions MPEG-7 au Schéma XML

En plus de ces deux composants spécifiques à XML, certaines extensions aux schémas XML ont dû être ajoutées afin de mieux répondre aux particularités des données multimédia. En particulier, un support pour les types de données tableaux et matrices, ou encore des types de données temporelles ont été ajoutés au schéma XML.

Puisque le DDL est une extension du Schéma XML, il peut être considéré comme un langage de définition de schéma à part entière, pour les documents XML de portée générale. De plus, les descriptions MPEG-7 constituent une instanciation des Schémas de Description définis dans le DDL. Par conséquent, une Description peut être considérée comme un

document XML, valide au regard de la DDL, et contenant le Schéma de Description du média auquel s'applique la description.

Dans une définition de schéma DDL MPEG-7, un Schéma de Description est défini au moyen d'un type complexe.
Un type complexe peut être référencé par une déclaration de type d'élément pour spécifier les éléments et les valeurs d'attribut pouvant apparaître comme les contenus des éléments instanciant le type d'élément déclaré.

II.3.2 – Le Visuel

Les outils de descriptions visuelles de MPEG-7 sont des structures de base ou des Descripteurs couvrant des caractéristiques visuelles de base telles que la couleur, la texture, la forme, le mouvement, la localisation et la reconnaissance de visage. Tous ces outils que nous allons aborder sont décrits dans [ISO03].

A – Structures de base

En ce qui concerne le Visuel, il y a cinq structures de bases. Il s'agit du Grid Layout, des séries temporelles, de la vue multiple, des coordonnées spatiales 2D et de l'interpolation temporelle.

Le Grid Layout

Le Grid Layout éclate l'image en plusieurs zones rectangulaires de même taille, de sorte que chaque zone puisse être décrite séparément. Leur description se fait à l'aide d'autres Descripteurs (par exemple la couleur ou la texture).

Le Descripteur des séries temporelles (*Time Series*)

Ce descripteur définit des séries temporelles de Descripteurs dans une séquence vidéo et fournit une image des fonctionnalités permettant de faire correspondre une image avec une image vidéo ou des images vidéo entre-elles. Les Séries Temporelles sont soit régulières, soit irrégulières. Dans le premier cas, les Descripteurs localisent à intervalles réguliers pendant un laps de temps donné. Dans le second cas, les Descripteurs localisent à intervalles variés pendant un laps de temps donné.

Le Descripteur de la vue multiple 3D en 2D (*2D-3D Multiple View*)

Le Descripteur 2D/3D permet de combiner des Descripteurs 2D pour représenter une caractéristique visuelle d'un objet en 3D à partir d'angles de vue différents. Tous les descripteurs visuels 2D peuvent être utilisés.

Le Descripteur des coordonnées spatiales 2D (*Spatial 2D Coordinates*)

Ce Descripteur définit un système de coordonnées spatiales à deux dimensions ainsi qu'une unité qui seront utilisés dans d'autres Ds et DSs lorsque cela sera nécessaire. Ce système de coordonnées est défini par une correspondance entre l'image et le système de

coordonnées. Cela permet de conserver la description MPEG-7 d'une image, même si la taille de celle-ci est modifiée. Dans ce cas, la seule chose requise est la description de la correspondance entre l'image originale et l'image éditée.

Le Descripteur d'interpolation temporelle (*Temporal Interpolation*)

Ce Descripteur peut être utilisé pour estimer les valeurs d'une variable multi-dimensionnelle qui changent dans le temps (par exemple la position d'un objet dans une vidéo).

B – Descripteurs de couleur

Les Descripteurs de couleur sont, entre autres, l'espace de couleurs, la quantification de la couleur, les couleurs dominantes, la couleur évolutive et la disposition des couleurs.

Le Descripteur de l'espace de couleurs (*Color Space*)

Ce descripteur permet de définir l'espace de couleurs utilisé dans les autres descriptions basées sur la couleur. Il s'agit notamment des espaces de couleurs RGB et YC_rC_b que nous avons abordés dans la section II.2.3, ou encore de l'espace de couleurs HSV[12].

Le Descripteur de la quantification de la couleur (*Color Quantization*)

Ce descripteur définit une quantification uniforme d'un espace de couleurs. Ce descripteur peut être combiné avec des descripteurs de couleurs dominantes pour expliquer, par exemple, le sens des valeurs des couleurs dominantes.

Le Descripteur des couleurs dominantes (*Dominant Colors*)

Ce descripteur convient surtout pour représenter des caractéristiques locales telles qu'un objet ou une zone d'image. En effet, pour ce genre d'éléments, une petite quantité de couleurs seulement est nécessaire pour décrire l'information qu'ils contiennent.

Le Descripteur de couleur évolutive (Scalable Color)

Ce descripteur est un histogramme de couleur dans l'espace de couleur HSV, qui est encodé à l'aide d'une transformée de Haar. Le Descripteur de couleur évolutive est utile pour la comparaison d'image et la recherche basée sur la couleur. La précision des résultats de la recherche augmente, évidemment, avec le nombre de bits utilisés pour la représentation.

[12] L'espace de couleur HSV (pour Hue Saturation Value) est un modèle de représentation des couleurs basés sur la *Teinte* (i.e. la nuance de couleur), la *Saturation* (i.e. la pureté de la teinte) et la *Valeur* (i.e. la quantité de lumière que répand une couleur).

La Disposition des Couleurs (*Color Layout*)

Ce descripteur représente, sous une forme compacte, la distribution spatiale de la couleur des signaux visuels.

C – Descripteurs de formes

Les Descripteurs de formes permettent notamment de décrire une forme en se basant sur les zones qui la composent ou sur son contour.

Le Descripteur de formes basé sur les zones (*Region Shape*)

Comme nous pouvons le voir à la figure II.12, la forme d'un objet peut être représentée soit par une zone (ou tache) unique, soit par un ensemble de zones ou de trous. Etant donné que le descripteur utilise chaque pixel constituant la forme dans une image, il peut décrire n'importe quelle forme. Cela signifie qu'il peut aussi bien décrire une simple forme constituée d'une seule zone (cf. figures II.12 (a) et (b)) que des formes complexes constituées de trous (cf. figure II.12 (c)) ou de plusieurs zones (cf. figure II.12 (d) et (e)).

(a) **(b)** **(c)** **(d)** **(e)**

Figure II.12. Exemples de formes. **(a)** et **(b)** Formes constituées d'une seule zone. **(c)** Forme constituée de trous.

Ce Descripteur peut considérer comme différents des objets qui sembleraient similaires pour des humains. A titre d'exemple, les images de la figure III.13 sont des représentations très similaires pour une tasse. Toutefois les poignées des premières tasses (figures II.13 (a) et (b)) sont creuses (i.e. constituées d'un trou), alors que celle de la troisième tasse (figure III.13 (c)) a une poignée pleine. Le Descripteur de formes basé sur les zones considère la représentation de cette troisième tasse différente de celles des deux premières tasses.

(a) **(b)** **(c)**

Figure II.13. Représentations similaires d'un objet. **(a)** et **(b)** sont considérées comme étant similaires pour le Descripteur de formes basé sur les zones alors que **(c)** est considérée comme étant différente des deux autres.

Le Descripteur de formes basé sur le contour (*Contour Shape*)

Ce descripteur capture les caractéristiques de la forme d'un objet ou d'une zone en se basant sur son contour. Il utilise une représentation qui capture très bien les caractéristiques de la forme, permettant la recherche basée sur la similarité. Cette représentation reflète les propriétés de la perception du système visuel de l'homme et est compacte.

E – Descripteurs de mouvement

Les Descripteurs de mouvement concernent notamment le mouvement de caméra et la trajectoire du mouvement.

Le Descripteur du mouvement de caméra

Ce descripteur caractérise les paramètres de mouvement d'une caméra 3D. Il se base donc sur l'information de ces paramètres qui peuvent être extraits de manière automatique, au moyen d'outils de capture.

Le descripteur de mouvement de caméra supporte les opérations de bases d'une caméra:

- Fixe.
- Rotation horizontale (*panning*).
- Mouvement transversal horizontal (*tracking*).
- Rotation verticale (*tilting*).
- Mouvement transversal vertical (*booming*).
- Changement de la focale (*zooming*).
- Translation le long de l'axe optique (*dollying*).
- Rotation autour de l'axe optique (*rolling*).

La figure II.14, issue de [ISO03], illustre ces différents mouvements de caméra.

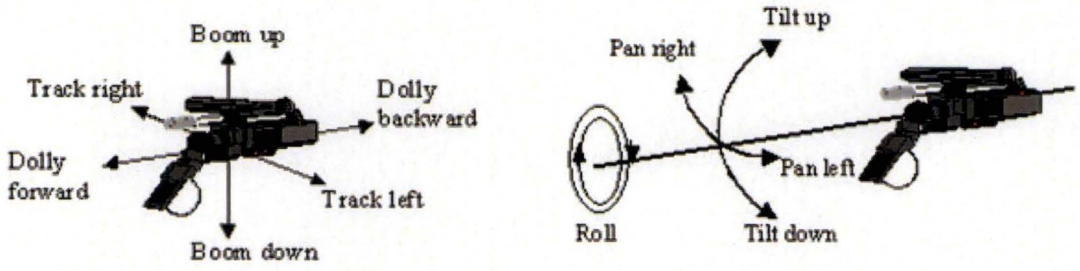


Figure II.14. Les mouvements de caméra.

Le Descripteur de trajectoire de mouvement

La trajectoire de mouvement d'un objet est une caractéristique de haut niveau simple. Il s'agit de la localisation dans le temps et dans l'espace d'un point représentatif de l'objet en question.
Ce descripteur consiste essentiellement en une liste de points-clés, et un ensemble de fonctions qui décrivent le chemin que suivent les objets entre ces points-clés.

F – Localisation

Il y a deux Descripteurs pour la localisation: Le délimiteur de zones et le délimiteur spatio-temporel.

Le délimiteur de zones

Ce descripteur permet de localiser des zones à l'intérieur d'une image, en les représentant par une boite ou un polygone.

Le délimiteur spatio-temporel

Ce descripteur décrit des zones spatio-temporelles dans une séquence vidéo (par exemple, des zones d'objets en mouvement) et fournit des fonctionnalités de localisation.

G – Reconnaissance de visages

Le descripteur de reconnaissance du visage peut être utilisé pour rechercher des images qui correspondent à la photographie d'un visage lors d'une requête.

II.3.2 – L'Audio

La partie Audio de MPEG-7 fournit des structures permettant de décrire le contenu audio. Elle propose une série de Descripteurs de bas niveau, pour les caractéristiques audio qui se retrouvent dans beaucoup d'applications. Ces dernières sont, par exemple, des caractéristiques temporelles, paramétriques et spectrales d'un signal. La partie Audio de MPEG-7 propose également une série de Descripteurs de haut niveau qui sont plus spécifiques à certaines applications. Il s'agit, par exemple, du Schéma de Description de la signature audio, des Schémas de Description pour le timbre d'un instrument de musique, les Descripteurs de mélodie, etc.

A – Les Descripteurs audio de bas niveau

Les Descripteurs audio de bas niveau sont au nombre de 17. On retrouve parmi eux un Descripteur de silence qui attache une sémantique simple de "silence" (i.e. pas de son signifiant) à une séquence audio. Les autres outils de descriptions de bas niveau sont largement décrits dans [ISO03].

B – Les Descripteurs audio de haut niveau

Les Descripteurs audio de haut niveau concernent la signature audio, le timbre de l'instrument de musique, la description de mélodie, l'indexation et la reconnaissance de sons généraux, et le contenu parlé.

Les Schémas de Description de la signature audio

Le Descripteur de signature audio est résumé dans le Schéma de Description `AudioSignature`. Il s'agit d'une représentation condensée d'un signal audio destiné à fournir un identifiant de contenu unique dans le but d'une identification automatique et robuste des signaux audio.

Les Outils de Description du timbre d'un instrument de musique

Les Descripteurs de timbre décrivent les caractéristiques des sons d'un instrument. Le timbre est la caractéristique qui fait que deux sons ayant la même tonalité et la même intensité semblent différents.

Les Outils de Description de mélodies

Le Schéma de Description de mélodies est une représentation compacte de l'information mélodique. Il sera abordé plus tard dans ce chapitre.

Les Outils de Description d'indexation et de reconnaissance de sons généraux

Les Outils de Description d'indexation et de reconnaissance générales du son consistent en une collection d'outils pour l'indexation et la classification de sons généraux.

Les Outils de Description du contenu parlé

Les Outils de Description du contenu parlé permettent une description détaillée des mots prononcés dans un flux audio.

II.3.3 – Les Schémas de Description Multimédia

MPEG-7 fournit des structures de métadonnées pour la description et l'annotation des contenus audiovisuels. Ces structures, appelées Schémas de Descriptions Multimédia (ou MDSs), permettent de décrire les concepts relatifs à la description de contenu audiovisuel et à la gestion de contenu. Cela permet de faciliter la recherche, l'indexation, le filtrage et les accès. Les Schémas de Description sont définis à l'aide de la DDL MPEG-7 sous la forme de documents ou de flux.

Les Schémas de Description Multimédia MPEG-7 sont organisés de la manière illustrée à la figure II.15.

Nous l'avons vu, les Descripteurs permettent de décrire des caractéristiques de bas niveau, telles que la couleur ou la forme d'une image. Nous le verrons dans la troisième partie de ce mémoire, ces caractéristiques peuvent être extraites des documents multimédias de manière automatique. Les Schémas de Description, quant à eux permettent de décrire des caractéristiques de plus haut niveau telles que des zones, des segments, des objets ou des événements. Ils produisent des descriptions beaucoup plus complexes en combinant plusieurs Descripteurs et Schémas de Description et en déclarant des relations entre les composants de la description.

Les Schémas de Description concernent les domaines du multimédia, de l'audio et du visuel. Alors que les DS audio ou visuels se réfèrent uniquement à des caractéristiques propres à leur domaine, les DS multimédia concernent une combinaison de données audio, visuelles et même textuelles. Dans certains cas, l'instanciation des Schémas de Description peuvent se faire de manière automatique, mais en général cela requiert une intervention extérieure (par un être humain).

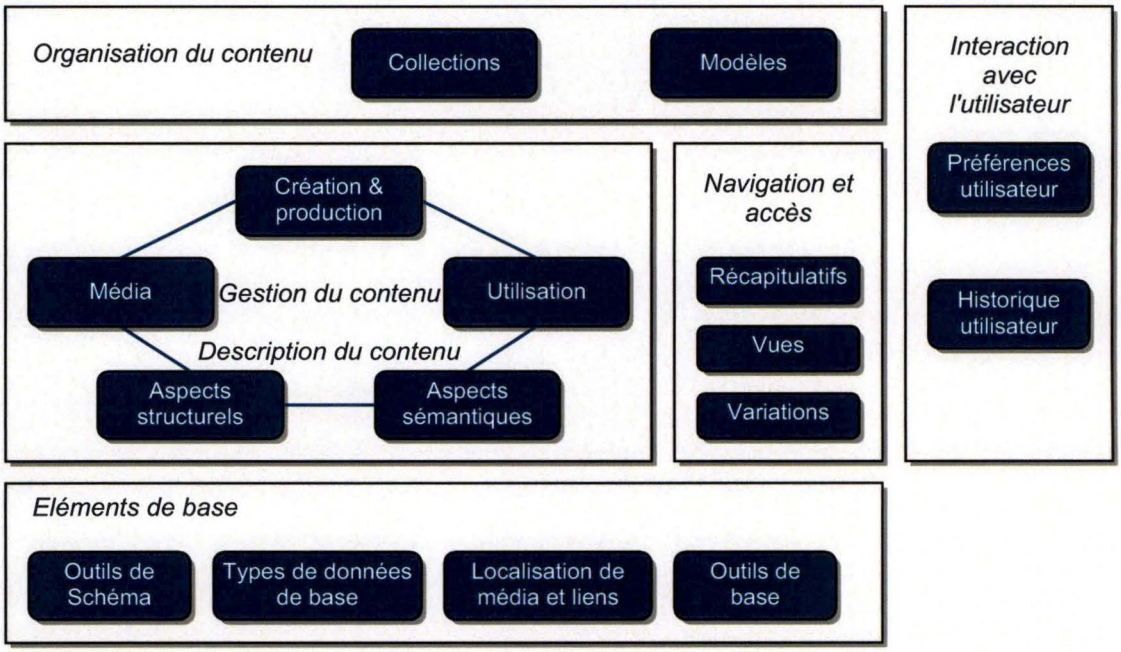


Figure II.15. Vue d'ensemble des Schémas de Description Multimédia MPEG-7.

A – Les éléments de base

Les Schémas de Description MPEG-7 sont définis à l'aide d'éléments de base. Les types de données de base fournissent une série de types de données étendus, et de structures mathématiques (vecteurs ou matrices). Ces éléments sont utilisés par les Schémas de Description pour décrire le contenu audiovisuel. Les éléments de base comprennent également des structures permettant de lier les fichiers média, localiser des parties de contenu, et de décrire différentes choses telles que le temps, le lieu, les personnes, les individus, les groupes, les organisations et d'autres annotations textuelles.

B – La gestion du contenu

Les Schémas de Description concernant la gestion du contenu fournissent de l'information sur les éléments suivants:

- La **création et production** qui consistent en de la méta information permettant de décrire la création et la production du contenu. Les caractéristiques concernent essentiellement le titre, le créateur, la catégorie, etc. Cette information est généralement produite par l'auteur puisqu'elle ne peut pas être générée à partir du contenu.
- Le **média** relatif à la description du média de stockage. Les caractéristiques concernent essentiellement le format de stockage, l'encodage du contenu

audiovisuel, les éléments permettant d'identifier le média. Il est possible de décrire plusieurs instances du média de stockage pour le même contenu audiovisuel.

- L'**utilisation** qui consiste en de la méta information relative à l'utilisation du contenu. Les caractéristiques concernent essentiellement les détenteurs de droit, les droits d'accès, la publication et l'information financière. Cette information peut changer pendant la durée de vie du contenu audiovisuel.

C – La description du contenu

Les Schémas de Description concernant la description du contenu permettent de décrire:

- Les **aspects structurels** qui concernent la description du contenu audiovisuel au point de vue de sa structure. La description est structurée en segments qui représentent des composants spatiaux, temporels ou spatiotemporels du contenu audiovisuel. Chaque segment peut être décrit à l'aide de caractéristiques basées sur le signal (la couleur, la texture, la forme, les mouvements, les caractéristiques audio) et quelque information sémantique supplémentaire.
- Les **aspects sémantiques** qui concernent la description du contenu audiovisuel au point de vue des ses notions conceptuelles. La description est structurée en objets, événements, concepts et diverses autres notions provenant du monde réel et qui sont capturées par le contenu audiovisuel.

Les Schémas de Description de contenu et de gestion sont inter-reliés et peuvent partiellement être inclus l'un dans l'autre dans les descriptions MPEG-7. Par exemple, l'information d'utilisation, la création et production de l'information sur les média peuvent être attachées à des segments individuels concernés dans la description structurelle du contenu. En fonction de l'application, certaines zones de la description du contenu audiovisuel pourront être accentuées (par exemple, la description des aspects sémantiques) et certaines autres minimisées ou ignorées (par exemple, la description des aspects structurels).

D – La navigation et l'accès

Les Schémas de Description concernant la navigation et l'accès facilitent le parcours et la recherche des contenus audiovisuels. A cette fin, ils définissent des récapitulatifs, des partitions et décompositions, ainsi que des variations du matériel audiovisuel.
Les récapitulatifs fournissent des résumés du contenu audiovisuel permettant notamment de parcourir et de visualiser le contenu audiovisuel. Les partitions et décompositions (ou vues) décrivent différentes décompositions des signaux audiovisuels dans l'espace, le temps et la fréquence. Elles peuvent être utilisées pour décrire différentes vues de la donnée audiovisuelle. Les variations, quant à elles, fournissent de l'information concernant différentes variations des programmes audiovisuels (résumés, et extraits, versions supportant différentes langues et modalités – audio, vidéo, image, texte, etc). L'un des buts visés par ce DS est de permettre la sélection de la variation la plus pertinente d'un programme audiovisuel, qui peut remplacer l'original pour s'adapter, par exemple, aux préférences de l'utilisateur.

E – L'organisation du contenu

Les Schémas de Description concernant l'organisation du contenu permettent d'organiser et de modéliser les collections de contenus audiovisuels et de descriptions.

Le Schéma de Descriptions Collection structure des collections de contenus audiovisuels, de segments, d'événements, et/ou d'objets. De cette façon, chaque collection peut être décrite comme un tout, à l'aide des propriétés communes. Le Schéma de Descriptions Model, quant à lui, fournit des outils permettant de modéliser les attributs et les caractéristiques du contenu audiovisuel.

F – L'interaction avec l'utilisateur

Les Schémas de Description concernant l'interaction avec l'utilisateur permettent de décrire les préférences utilisateur et l'historique d'utilisation du matériel multimédia. Cela permet notamment de faire correspondre les préférences utilisateur avec les descriptions de contenu MPEG-7 afin de faciliter la personnalisation de l'accès au contenu audiovisuel, sa présentation et sa consommation.

En guise de résumé, le tableau II.7, que l'on peut retrouver dans [UTZ04], nous donne un aperçu des Schémas de Description visuels, audio et multimédia.

Tableau II.7. Schémas de Descriptions MPEG-7 prédéfinis.

Standardized MPEG-7 media description schemes		
Visuel	Audio	**Multimédia**
Color: ▪ color space ▪ dominant colors ▪ color quantization ▪ ...	Audio framework: ▪ audio waveform ▪ audio power ▪ audio spectrum ▪ ...	Content management: ▪ creation information ▪ creation tool ▪ creator ▪ ...
Texture: ▪ edge histogram ▪ homogeneous texture ▪ texture browsing ▪ ...	Timbre: ▪ harmonic instrument timbre ▪ percussive instrument timbre ▪ ...	Content semantics: ▪ classification scheme ▪ text annotation ▪ graph ▪ ...
Shape: ▪ object region-based shape ▪ contour-based shape ▪ 3D shape ▪ ...	Sound recognition and indexing: ▪ sound model ▪ sound classification model ▪ sound model state path ▪ ...	Navigation and summarization: ▪ hierarchical summary ▪ visual summary component ▪ audio summary component ▪ ...
Motion: ▪ camera motion ▪ object motion trajectory ▪ motion activity ▪ ...	Melody: ▪ melody contour ▪ melody sequence ▪ ...	Content organization: ▪ collection ▪ classification model ▪ cluster model ▪ ...
Localization: ▪ region-locator ▪ spatio-temporal locator ▪ ...	Spoken content: ▪ spoken content lattice ▪ spoken content header ▪ ...	User interaction: ▪ usage history ▪ user preferences ▪ ...

Une caractéristique importante des types complexes utilisés par MPEG-7 est que les uns peuvent être dérivés des autres (soit par extension du modèle de contenu d'un type de base avec plus d'éléments ou valeurs d'attribut, soit par restriction du modèle de contenu du type de base à un sous-ensemble limité).
Chaque Schéma de Description est dérivé directement ou indirectement du type complexe `DSType` prédéfini par MPEG-7 (ou du type complexe `DType`, dans le cas où il s'agit d'un Descripteur). De cette manière, les schémas de description de média MPEG-7 sont organisés en une hiérarchie de dérivation profonde. Cela permet une combinaison flexible des différents Schémas de Description.

La figure II.16, reprise de [UTZ04], illustre la définition de Schémas de Description avec le DDL. Elle montre une simplification du Schéma de Description de mélodies, représentatif de la description de mélodies et pouvant être utilisé dans un contexte d'applications de requêtes par chantonnement.

```
...

<complexType name="MelodyType">
  <complexContent>
    <extension base="mpeg7:AudioDSType">
      <sequence>
        <element name="Meter"
          type=" mpeg7:MeterType"
          minOccurs="0"/>
        <element name="MelodyContour"
          type=" mpeg7:MelodyContourType"
          minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="MelodyContourType">
  <complexContent>
    <extension base="mpeg7:AudioDSType">
      <sequence>
        <element name="Contour">
          <simpleType>
            <list itemType="integer"/>
          </simpleType>
        </element>
        <element name="Beat">
          <simpleType>
            <list itemType="integer"/>
          </simpleType>
        </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="MeterType">
  <complexContent>
    <extension base="mpeg7:AudioDType">
      <sequence>
        <element name="Numerator">
          <simpleType>
            <restriction base="integer">
              <minInclusive value="1"/>
              <maxInclusive value="128"/>
            </restriction>
          </simpleType>
        </element>
        <element name="Denominator">
          <simpleType>
            <restriction base="integer">
              <enumeration value="1"/>
              <enumeration value="2"/>
              <enumeration value="4"/>
              <enumeration value="8"/>
              <enumeration value="16"/>
              <enumeration value="32"/>
              <enumeration value="64"/>
              <enumeration value="128"/>
            </restriction>
          </simpleType>
        </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="AudioDescriptionScheme"
  type="mpeg7:AudioDSType"/>

...
```

Figure II.16. Schéma de Description de mélodies.

Une mélodie est définie par le type complexe *MelodyType*, dans la partie supérieure de la colonne de gauche de la figure. En étendant le type prédéfini *AudioDSType* qui, à son tour, étend le type prédéfini `DSType`, il en ressort que *MelodyType* définit un Schéma de Description audio MPEG-7.

La déclaration de *MelodyType* exprime qu'une mélodie peut être décrite par sa mesure et son contour de mélodie en utilisant des éléments de *Meter* et *MelodyContour*.
La mesure d'une mélodie selon le type complexe *MeterType* (partie supérieure de la colonne de droite de la figure), fournissant le modèle de contenu pour les éléments *Meter*, est une fraction consistant en un numérateur et un dénominateur (types d'élément *Numerator* et *Denominator*). Ici, il y a une restriction selon laquelle le numérateur doit être une valeur entière comprise entre 0 et 128, alors que le dénominateur doit être une puissance de deux dans le même intervalle.

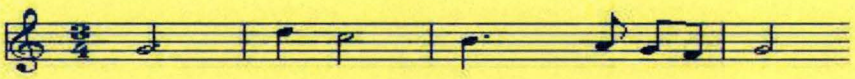
Un contour de mélodie est composé d'un contour et d'un battement. Il est exprimé à l'aide des types d'élément *Contour* et *Beat*, donnés par le type complexe *MelodyContourType* (partie inférieure de la colonne de gauche de la figure), qui définit le modèle de contenu pour les éléments *MelodyContour*. *MelodyContourType* étant dérivé de *AudioDSType*, il définit également un Schéma de Description Audio, c'est-à-dire le Schéma de Description *MelodyContour*.
Le contour d'une mélodie est une liste de valeurs entières donnant une mesure pour la distance entre deux notes consécutives. Le battement, quant à lui, constitue une liste de valeurs entières associant chaque note avec sa position dans le battement.
Nous pourrions également imaginer que la mélodie soit décrite à l'aide d'un troisième élément qui serait la clef de la mélodie. Dans ce cas, cet élément pourrait être déclaré comme suit:

```
<element name="key" type="string" minOccurs="0" maxOccurs="1">
```

Les instances suivantes de cet élément seraient alors valides:

`<key>A</key>` où A représente la note la
`<key>B</key>` où B représente la note si
`<key>C</key>` où C représente la note do
`<key>D</key>` où D représente la note ré
`<key>E</key>` où E représente la note mi
`<key>F</key>` où F représente la note fa
`<key>G</key>` où G représente la note sol

Enfin, la figure montre la déclaration du type d'élément *AudioDescriptionScheme*, dans la partie inférieure de la colonne de droite. Les contenus valides pour des éléments de ce type respectent le type complexe *AudioDSType*. Etant donné que chaque Schéma de Description audio prédéfini par MPEG-7 est dérivé de *AudioDSType*, les éléments *AudioDescriptionScheme* peuvent contenir des Descriptions satisfaisant à n'importe quel Schéma de Description audio. En particulier, un élément *AudioDescriptionScheme* peut être rempli conformément aux Schémas de Description *Melody* et *MelodyContour* si des valeurs d'attribut `xsi:type` sont fournies, s'adressant aux types complexes *MelodyType* et *MelodyContourType*, respectivement.



```
<!-- Description de mélodie de 8 notes prises de "Moon River" de Henri Mancini -->

<AudioDescriptionScheme xmlns="http://www.mpeg7.org/..."
                        xmlns:xsi="http://www.w3.org/..."
                        xsi:type"MelodyType">
    <Meter>
        <Numerator>3</Numerator>
        <Denominator>4</Denominator>
    </Meter>
    <MelodyCountour>
        <!-- Distance entre deux notes -->
        <Countour>2 -1 -1 -1 -1 1</Countour>
        <!-- Position de battement des notes -->
        <Beat>1 4 5 7 8 9 9 10</Beat>
    </MelodyCountour>
</AudioDescriptionScheme>
```

Figure II.17. Exemple de Description MPEG-7.

La figure II.17 illustre une description de média MPEG-7 satisfaisant au Schéma de Description de mélodies. La description couvre une petite fraction de la mélodie de la chanson « Moon River » de Henri Mancini. Un élément *AudioDescriptionScheme* constitue le point d'entrée de la description dont le contenu est marqué par une valeur d'attribut `xsi:type` pour complaire au Schéma de Description de mélodies.
Nous pourrions ajouter la ligne suivante, satisfaisant à la déclaration de l'élément *key* de tout à l'heure:

```
<Key>G</Key>
```

Nous l'avons vu, les Schémas de Description permettent la description du contenu multimédia ou de parties de celui-ci, non seulement à un niveau sémantique mais aussi à un niveau technique. De ce fait, il fournit un support adéquat pour la représentation de métadonnées techniques plus complexes (par exemple, les histogrammes de couleur ou les formes). De plus, MPEG-7 peut être adapté aux besoins d'une application ou d'un domaine d'application particulier au moyen de la DDL.

Nous avons abordé, dans le premier chapitre, la Gestion Electronique de Documents et nous avons de bonnes bases concernant les documents multimédia. Nous avons donc de quoi introduire le thème de la gestion des données multimédia.

II.4 – La gestion de données multimédia

Cette section a pour but d'expliquer l'idée de l'application de la GED aux données multimédia. En effet, nous pouvons aisément imaginer d'utiliser les Descriptions de contenus MPEG-7 pour la recherche de documents.

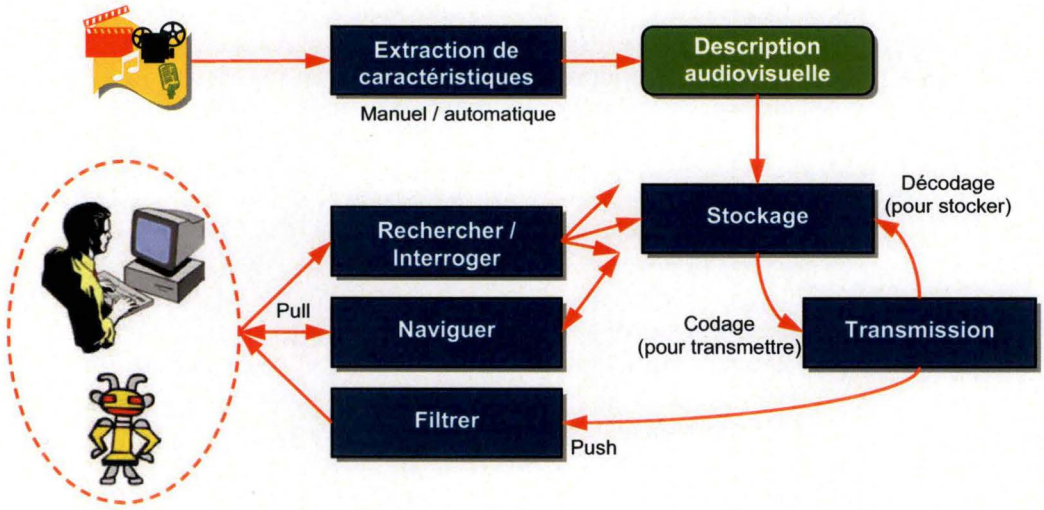


Figure II.18. Stockage et recherche de Description audiovisuelle.

La figure II.18, que l'on peut retrouver dans [ISO03], illustre un scénario de stockage et de recherche de Description Audiovisuelle.

Cette figure montre que l'on peut obtenir une Description audiovisuelle à partir d'un contenu multimédia, par extraction automatique ou semi-automatique. De là, la Description peut être stockée ou écoulée directement. Si l'on considère un scénario de retrait (pull), les applications clientes vont interroger le conteneur de Descriptions, et recevront une série de Descriptions correspondant à leur requête. Dans l'optique d'un scénario d'insertion (push), un filtre (par exemple, un agent intelligent) sélectionnera les descriptions parmi celles disponibles et accomplira des actions programmées (par exemple, un changement de canal de diffusion), par après. Quel que soit le scénario, chaque module peut manipuler des Descriptions codées aux formats MPEG-7 (textuel ou binaire).

Si l'on dispose de moyens pour retrouver la Description du contenu d'un document audiovisuel, il suffirait de la lier au document multimédia, pour obtenir un outil de recherche pour celui-ci.
Un tel outil serait intéressant, par exemple dans le domaine de la gestion documentaire multimédia. Pensons notamment à la télévision nationale qui a accumulé une quantité impressionnante d'archives audiovisuelles. La possibilité de retrouver aisément un document donné à l'aide d'un outil de recherche approprié, réduirait considérablement les ressources en temps et hommes nécessaires. Un autre exemple est celui de musées qui doivent également gérer de vastes collections de documents multimédia.
Imaginons enfin un scénario dans lequel les internautes auraient la possibilité de louer sur le Web les films de leur choix. Un tel système leur permettrait de retrouver plus facilement le film désiré, même s'ils n'en connaissent pas spécialement le titre. Pensons, par exemple à un fan de Jacques Brel incapable de retenir le titre des films dans lesquels il a joué. Il lui suffirait d'effectuer une requête telle que "retrouvez, s'il vous plait, tous les films dans lesquels joue Jacques Brel", pour récupérer les vidéos en question.

Les moteurs de recherche classiques tels que Alta Vista proposent de décrire les images en utilisant des mots-clés correspondant aux attributs extérieurs au document. Ces attributs sont, par exemple le nom du document ou le texte alternatif à l'image dans le code HTML. Ce type d'indexation se base donc sur le profil du document, c'est-à-dire des informations externes au document telles que son nom, le nom de l'auteur, la date de publication, etc.

Une autre solution est de représenter le document par des descripteurs obtenus à partir de son contenu. C'est ce que l'on appelle communément la recherche par (ou basée sur) le contenu. La description du document peut être faite de façon manuelle ou automatique.
L'avantage de la description manuelle est que l'on peut représenter directement le document par des mots, ce qui facilite la recherche par mots-clés. Toutefois, l'indexation des documents étant souvent réalisée par plusieurs personnes, il y a un risque que deux documents similaires ne soient pas décrits avec les mêmes termes, ce qui rendrait les index incohérents. Un autre inconvénient est qu'il n'est pas toujours facile de décrire une image avec des mots. Pensons, par exemple à la disposition spatiale des zones de couleurs contenue dans une image. C'est pourquoi on a souvent recours à des techniques permettant d'extraire les caractéristiques d'un document multimédia pour indexer celui-ci ou pour retrouver d'autres documents similaires.

La deuxième partie de ce mémoire aborde un système de recherche de documents multimédia par mots-clés. Dans ce système, on suppose qu'il existe des fichiers décrivant le contenu du document. Mais, comme nous venons de le voir, il n'est pas toujours pertinent d'avoir recours à la recherche par mots-clés et à la description manuelle, dans le cas des documents multimédia. C'est pourquoi la troisième partie de ce mémoire se propose de faire un tour d'horizon de la recherche par le contenu.

Partie 2

Un système de recherche par mots-clés: Le Data Manager

CHAPITRE TROISIEME – DESCRIPTION DU DATA MANAGER

Lors de mon stage de fin d'études, effectué au sein de la société Arafox, j'ai eu l'occasion de participer au développement d'un projet appelé "Data Manager".
Arafox est une petite société qui fut créée au début de l'année 2000 et qui se spécialise dans les solutions GNU / Linux et dans les développements de logiciels libres. Les travaux de la société se concentrent autour de quatre domaines: l'ingénierie des systèmes et réseaux GNU / Linux, le développement de logiciels en Open-Source, les formations professionnelles ainsi que le conseil et le support.
Toutes les tâches et activités de l'entreprise s'articulent autour du libre ce qui a l'avantage pour le client d'être peu onéreux. De plus, le code source des logiciels étant disponible, et donc facile d'adaptation, les programmes ont une durée de vie beaucoup plus longue.

Le Data Manager s'inscrit dans le cadre d'un projet de système d'information documentaire multimédia, pour le Musée Albert Kahn (Paris). Ce projet a pour finalité l'édition, la recherche et la consultation de documents multimédia. C'est pourquoi le système possède une architecture client-serveur et est organisé dans un réseau à haut débit.
Le noyau de ce système, le Data Manager, est un gestionnaire de méta-données. Il sert de stockage et de distribution de documents multimédia contenant des descripteurs XML et les documents audiovisuels à proprement parler. A cette fin, les fonctionnalités qu'il fournit permettent la recherche par mots-clés et par approximation orthographique sur les documents qu'il contient, tout en prenant en compte les synonymes. Le Data Manager s'articule donc autour d'une base de données XML native et intègre un moteur d'indexation ainsi qu'un gestionnaire de synonymes.

La figure III.1 illustre le fonctionnement général du Data Manager. Imaginons, par exemple, que nous cherchions la photo d'une petite fille portant un t-shirt rouge. A chaque image contenue dans le système, correspond une description au format XML, qui a été indexée et stockée lors du stockage de la photo. Le stockage d'un document audiovisuel se fait dans une banque de données de type WebDAV alors que sa description est stockée dans une base de données XML native et indexée à l'aide d'un moteur d'indexation. De cette façon, si la photo que nous cherchons se trouve dans le système, sa description aura été indexée et il nous est donc possible de la retrouver. A cette fin, nous pouvons interroger le système, par l'intermédiaire du moteur de recherche.
Pour récupérer la photo désirée, il nous est possible d'interroger le système, par l'intermédiaire de requêtes adressées à l'indexeur. Une telle requête serait, par exemple, "recherchez, s'il vous plait, toutes les photos disponibles sur lesquelles on peut voir une fillette et un t-shirt rouge". Le système nous répondra en fournissant les descriptions des documents répondant à ce critère. Sur base de celles-ci, il nous sera possible de sélectionner les images répondant le plus à notre attente. A partir de là, les documents correspondants pourront être récupérés dans la base de données.
Afin d'obtenir les mêmes résultats, que l'on utilise le terme "fillette" ou "petite fille", le système prend en compte les synonymes des mots constituant la requête. En outre, pour les utilisateurs distraits qui oublieraient un "t" à "fillette", le Data Manager vérifie l'orthographe des différents mots de la requête.

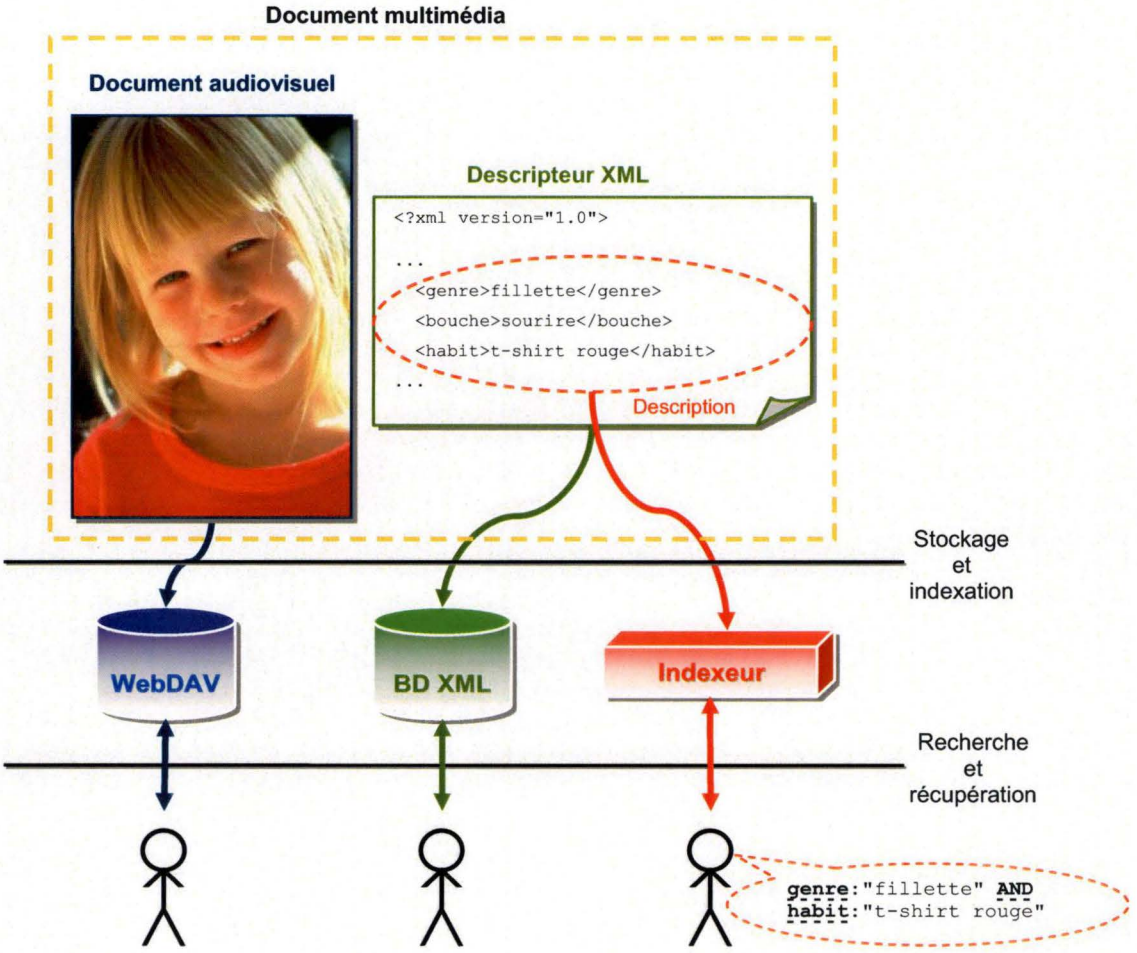


Figure III.1. Fonctionnement général du Data Manager.

III.1 – Une base de données XML native

Le Data Manager dispose d'une base de donnée XML native, pour le stockage et la gestion des descriptions XML.
Dans cette section, nous introduirons, dans un premier temps les notions de bases de données XML. Ensuite nous aborderons trois bases de données XML natives Open Source qui sont dbXML, eXist et Xindice. Celle qui est utilisée dans le Data Manager est la base eXist.

III.1.1 – Les bases de données XML

A mesure que XML gagnait en popularité en tant que langage de description et d'échange de documents structurés, divers outils nécessaires au traitement et au stockage ont vu le jour. Il s'agit notamment des bases de données XML. En réalité, il existe deux types de solutions répondant au problème de gestion de documents XML dans une base de

données: les extensions des bases de données relationnelles et les bases de données XML natives (ou NXDs pour native XML databases). Les sections suivantes fournissent une introduction à ces solutions.

A – Les bases de données relationnelles étendues

Globalement, selon [UTZ04] et [VAR02], on peut distinguer plusieurs approches pour représenter les documents XML dans une base de données relationnelle:

- Le *stockage non structuré* qui consiste à stocker le document en bloc dans le champ d'une table.
- Le *stockage structuré* qui consiste à extraire l'information contenue dans le document pour l'éclater dans différentes tables et champs.
- Le *mapping* qui consiste à mapper le contenu d'un document avec des schémas de bases de données créés spécifiquement pour ce contenu.

Dans les bases de données relationnelles répondant à la première approche, le document XML est stocké sous sa forme textuelle, dans un CLOB (pour character large object). Aujourd'hui, la plupart des bases de données relationnelles ont été étendues afin de supporter le stockage non structuré de documents XML. Cette approche permet de préserver le document dans son intégrité, mais n'offre pas une manipulation facile des données à l'intérieur du document.
Les systèmes de gestion de bases de données relationnelles (SGBDR) utilisant la seconde approche construisent un méta-modèle de documents XML permettant la représentation en arbre de documents XML. Cela permet aux fonctions de recherche d'accéder à la structure et aux contenus des documents XML. Cette approche ne permet pas de conserver le document original intact.
Enfin, les bases de données relationnelles répondant à la troisième approche mettent toutes les capacités de modélisation d'un SGBDR à disposition afin de représenter le contenu du document de manière efficace et adéquate.

B – Les bases de données XML natives

Selon l'Initiative XML:DB, une base de donnée XML native peut être définie de la manière suivante[1]:

- *"Une base de données XML native définit un modèle (logique) de documents XML – modèle est ici opposé aux données du document – et stocke et retrouve les documents en fonction de ce modèle. Le modèle doit au minimum inclure les éléments, les attributs, les PCDATA et l'ordre interne du document. Quelques exemples de tels modèles sont le modèle de données de XPath, le glossaire XML Infoset, et les modèles implicites de DOM et des événements de SAX 1.0.*
- *Le document XML est l'unité fondamentale du stockage[2] (logique) dans une base de données XML native, tout comme une ligne d'une table constitue l'unité fondamentale de stockage (logique) dans une base de données relationnelle.*
- *Une base de données XML native ne repose pas sur un modèle physique particulier pour le stockage. Elle peut, par exemple être bâtie aussi bien sur une base relationnelle hiérarchique, orientée-objet, ou bien utiliser des techniques de stockages propriétaires comme des fichiers indexés ou compressés."*

[1] On peut également trouver cette définition dans [BOU04] ou encore [STA01].
[2] L'unité fondamentale de stockage correspond au niveau le plus bas du contexte dans lequel un lot de données peut s'adapter.

Il existe deux catégories de bases de données XML natives: l'une regroupant les NXDs offrant un stockage basé sur le texte, l'autre les NXDs supportant un stockage basé sur le modèle.
Les NXDs de la première catégorie stockent le document sous la forme d'un texte et fournissent plusieurs fonctionnalités permettant l'accès au document. Cela permet d'assurer l'intégrité du document et la rapidité de traitement.
Les NXDs de la seconde catégorie, quant à elles, stockent un modèle binaire du document (par exemple DOM) dans une banque de données. Ici l'intégrité du document est assurée, seulement à hauteur du modèle de document sous-jacent. L'avantage de ce type de base de données XML native est qu'elle peut combiner des fragments de différents documents. Toutefois, le stockage basé sur le modèle tire un désavantage dans sa rapidité de traitement.

Toutes les NXDs sont différentes. Toutefois, elles ont des caractéristiques communes telles que les collections, les requêtes et les mises à jour. Les paragraphes suivants reprennent toutes les caractéristiques (ou presque) que possède une base de données XML native.

Les collections

Les bases de données XML natives gèrent des collections de documents. Une collection de documents est ce qui correspond à une table dans une base de données relationnelle. Elle permet de chercher et de manipuler des documents.
Supposons que nous voulions utiliser une NXDs pour stocker les coordonnées des artistes de la chanson française. Il faudrait alors créer une collection qui s'appellerait *artistes*, par exemple. Ainsi, les recherches que nous voudrions effectuer sur ces artistes seraient limitées aux documents de cette collection.
Si, par exemple, nous voulions stocker tous les titres des albums de chanson française, il nous faudrait décrire une collection pour chaque album, contenant chacune des collections correspondant aux différents titres. Cette structure hiérarchique des collections NXDs est prise en charge dans certaines NXDs.
Certaines NXDs, dites indépendantes du schéma, ne requièrent pas qu'un schéma soit associé à une collection. De cette façon, il est possible de stocker n'importe quel document dans la collection, quel que soit le schéma. Ce type de base de données XML native est, dès lors, extrêmement flexible et rend le développement d'applications plus facile.

Les requêtes

Certains NXDs supportent un ou plusieurs langages de requêtes, d'autres n'en possèdent pas. Le langage de requêtes le plus connu pour les NXDs est XPath[3]. Pour être à la hauteur d'un langage de requêtes de base de données, XPath est étendu. Ainsi, il permet des requêtes à travers des collections de documents, pour les recherches sur des documents multiples.
Selon [STA01], XPath possède des limites qui résident dans l'absence de groupement, de tri, de jointure (entre documents) et de support pour les types de données. Une partie de ces problèmes peut trouver une solution dans l'utilisation de XSLT ([XSLT]), mais ils peuvent être résolus avec un langage beaucoup plus orienté base de données, appelé XQuery ([XQUERY]).

[3] Vous pourrez trouver la recommandation W3C concernant XPath dans [XPATH]

Les mises à jour

La plupart des bases de données XML natives demandent à l'utilisateur de récupérer le document afin de le mettre à jour en dehors du système, "à la main". L'utilisateur utilise ainsi l'API XML qu'il désire pour effectuer les changements nécessaires avant de retourner le document dans la NXD. D'autres produits permettent de modifier un document à l'intérieur du système. Certains, en fournissant un langage propriétaire de mise à jour, d'autres (des NXDs Open Source) en supportant XUpdate de XML:DB.

Les transactions, verrouillages et accès concurrentiels

Toutes les NXDs (ou presque) supportent les transactions. En ce qui concerne le verrouillage, il se fait généralement au niveau du document, plutôt qu'au niveau du fragment de document. Ceci a pour conséquence de ralentir les accès multi-utilisateurs concurrentiels.

Les APIs

Les APIs (pour Application Programming Interfaces) des NXDs offrent des méthodes permettant à l'utilisateur de se connecter à la base, d'explorer les métadonnées, d'exécuter des requêtes et de rechercher des résultats. Ceux-ci consistent en une chaîne XML, un arbre DOM ou un analyseur SAX.

L'aller-retour des documents

Cette caractéristique permet de récupérer le même document que celui qui a été stocké. Cela signifie que les documents stockés dans la base de données XML native pourront être obtenus à nouveau sous leur forme "originale".

Les index

La majorité des NXDs permettent d'indexer des valeurs d'éléments et des attributs, afin de faciliter et d'accélérer les recherches.

Il existe deux grandes familles de bases de données XML native[4]: les solutions commerciales et les solutions libres que sont dbXML, eXist et Xindice. Les NXDs Open Source seront abordées dans les sections III.1.2 à 4. Etant donné que le Data Manager s'inscrit dans le cadre exclusif du logiciel libre, les solutions propriétaires ne seront pas abordées dans ce mémoire.

C – Pourquoi une base de données XML native?

Il existe diverses raisons pour lesquelles il est préférable de stocker les descriptions XML dans une NXDs plutôt que dans une base de donnée relationnelle.
La première est que la structure des données contenues dans le descriptif varie d'un document multimédia à l'autre. Nous pouvons aisément imaginer que la description d'une photo représentant un garçon roulant à vélo et celle d'une photo représentant un pommier au printemps n'auront pas la même structure. Comme nous pouvons le voir à la figure III.2, les

[4] En réalité il existe une troisième famille incarnée par les prototypes de recherche.

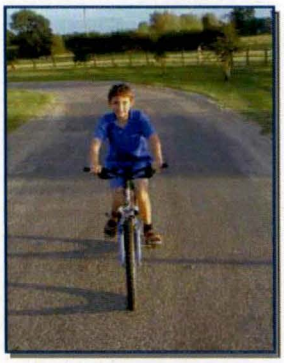
éléments décrivant la première photo ne sont pas les mêmes que ceux décrivant la deuxième photo. On pourrait transférer les données de tous les descriptifs de photos disponibles dans une seule table (appelée ici **PHOTOS**). Dans cette dernière, chaque colonne correspondrait à un nom d'élément possible de descriptif de photo. Il y aurait donc autant de colonnes que de noms d'éléments différents parmi tous les descriptifs de photo du système. Dans l'illustration, le système contient deux photos. La première est décrite à l'aide des éléments **plante**, **lieu** et **saison**, alors que les éléments décrivant la seconde photo sont **personne**, **véhicule** et **lieu**. Etant donné que le seul élément commun des descriptions ont est **lieu**, les colonnes de la table seraient alors **PLANTE**, **PERSONNE**, **VEHICULE**, **LIEU** et **SAISON** (ainsi, bien entendu que la colonne identifiante NPHOTO). En conséquence de cela, les champs **PERSONNE** et **VEHICULE** de la ligne concernant la première photo auraient une valeur nulle. De la même façon, les champs **PLANTE** et **SAISON** de la ligne concernant la deuxième photo auraient une valeur nulle.
Une autre solution consisterait à créer autant de tables qu'il y a de photos de types différents. Cela correspondrait, dans l'illustration à une table **CONDUCTEUR** pour les descriptions contenant les éléments **personne**, **véhicule** et **lieu** et une table **VEGETAL** pour celles contenant les éléments **plante**, **lieu** et **saison**. Cela éliminerait le problèmes des valeurs nulles, mais augmenterait considérablement le nombre de tables dans le système.
Enfin, une troisième solution consisterait à créer une table unique reprenant chaque nom d'élément du document et sa valeur associée. Cela correspond, dans l'illustration à la table **ELEMENT** où la colonne **NOM** contient le nom de l'élément et la colonne **VALEUR** contient la valeur associée. Une colonne supplémentaire **IDDOCUMENT** est nécessaire pour préciser de quel document provient l'élément. Ainsi, dans l'exemple, les trois premiers éléments de la table proviennent de la première photo (pic1) et les trois suivants proviennent de la seconde photo (pic2). L'inconvénient de cette solution est que la table contiendra beaucoup de lignes apportant chacune très peu d'information.
En conclusion, le transfert des données dans une base relationnelle peut conduire à:

- Un nombre important de colonnes à valeur nulle.
- Un nombre important de tables.
- Une dégradation de la structure de données.

Cela entraîne donc soit une perte de place, soit une perte d'efficacité.

Une deuxième raison à l'utilisation d'une NXD est que sa méthode de stockage permet de retrouver plus facilement et plus rapidement les données que ça ne l'est possible dans une base de données relationnelle. En effet, la base de données XML native sauvegarde physiquement des documents entiers ensemble ou lient les différentes parties des documents à l'aide d'un pointeur physique. Cela évite de devoir utiliser des techniques telles que la jointure logique des bases relationnelles, pour retrouver des documents.
A titre d'exemple, la figure III.3 montre que le document LittlegirlDescription.xml pourrait être stocké dans une base de données relationnelle en utilisant quatre tables: **PHOTO**, **VISAGE**, **PERSONNAGE** et **HABIT**. Donc, la recherche du document en question nécessiterait de joindre ces quatre tables. Il faudrait quatre consultations d'index et au moins quatre lectures pour retrouver le document ou une partie de celui-ci. En revanche, dans une base de données XML native, ce document XML pourrait être stocké à un emplacement unique sur le disque. De cette façon, lorsque l'on voudra rechercher le document ou une partie de celui-ci, il faudrait interroger une seule fois l'index et il faudrait qu'une seule lecture pour retrouver les données.
Nuançons cependant notre propos: le gain de vitesse s'applique seulement lorsque l'on recherche des données dans l'ordre où elles sont stockées sur disque. Si la recherche des données se fait selon une vue différente (par exemple lister les personnages de genre fillette apparaissant sur une photo de type JPEG), les performances seront alors probablement pires qu'avec une base relationnelle. Donc, dans ce cas, le stockage des données dans une NXD ne se justifie pas pour des raisons de performance.

Une troisième raison à l'utilisation de NXDs, plutôt que d'un SGBD pour le stockage des descriptions est que cela permet d'exploiter des fonctionnalités spécifiques à XML comme l'exécution de requêtes XML.



```
...
  <personne>
    garçon
  </personne>

  <véhicule>
    bicyclette
  </véhicule>

  <lieu>
    rue
  </lieu>
...
```

```
...
  <plante>
    pommier
  </plante>

  <saison>
    printemps
  </saison>

  <lieu>
    pré
  </lieu>
...
```

PHOTO					
NPHOTO	PLANTE	PERSONNE	VEHICULE	LIEU	SAISON
P25	Pommier	-	-	Pré	Printemps
P36	-	Garçon	Bicyclette	Rue	-

ou

CONDUCTEUR			
NC	PERSONNE	VEHICULE	LIEU
C12	Garçon	Bicyclette	Rue

et

VEGETAL			
NV	PLANTE	LIEU	SAISON
V24	Pommier	Pré	Printemps

ou

ELEMENT			
NELT	IDDOCUMENT	NOM	VALEUR
E28	PIC1	personne	garçon
E43	PIC1	véhicule	bicyclette
E53	PIC1	lieu	rue
E67	PIC2	plante	pommier
E68	PIC2	saison	printemps
E93	PIC2	lieu	pré

Figure III.2. Transfert des données XML dans une base de données relationnelle.

```
<photo>
   <proprietes>
      <titre>Little girl</titre>
      <type>JPEG</type>
      <adresse>D://pictures/personnage/littlegirl.jpg</adresse>
      <taille unite="octet">43993</taille>
      <dimension unite="pixel">283x421</dimension>
      <creation>20040308</creation>
      <modification>20040308</modification>
   </proprietes>
   <description>
      <personnage>
         <genre>fillette</genre>
         <visage>
            <yeux>bleux</yeux>
            <cheveux>blonds</cheveux>
            <bouche>sourire</bouche>
            </visage>
         <habit>
            <type>t-shirt</type>
            <couleur>rouge</couleur>
         </habit>
      </personnage>
   </description>
</photo>
```

LittlegirlDescription.xml

PHOTO

NPIC	TITRE	TYPE	ADRESSE	TAILLE	DIM	CREATION	MODIF
PC33	Bicycle	JPEG	D://pictures/conducteur/bicycle.jpg	27013	316x421	8/03/2004	8/03/2004
PP01	Littlegirl	JPEG	D://pictures/personnage/littlegirl.jpg	43993	283x421	8/07/2004	8/07/2004
PV12	Appletree	JPEG	D://pictures/vegetal/appeltree.jpg	222220	400x300	8/03/2004	8/03/2004
...	...	...	...	...	...	...	...

VISAGE

NVIS	YEUX	NEZ	CHEVEUX	BOUCHE	OREILLES
V012	bleus	-	blonds	sourire	-
V023	-	-	bruns	sourire	-
...	...	...	...		...

PERSONNAGE

NPERS	NPIC	NVIS	GENRE
PFO3	PC33	V012	garçon
PG12	PP01	PP01	fillette
...	...	...	...

HABIT

NHABIT	NPIC	TYPE	COULEUR
H23	PP01	t-shirt	rouge
H45	PC33	t-shirt	bleu
H112	PC33	short	bleu
H134	PC33	sandales	brunes
...	...	...	...

Figure III.3. Stockage d'un document XML dans une base de données relationnelle.

Les bases de données XML natives permettent de préserver autant les commentaires, les PIs[5] et les DTDs, que la structure physique du document. Bien que les bases de données étendues puissent le faire en théorie, ce n'est généralement pas le cas.

[5] Pour Processing Instructions, qui permettent de passer des instructions aux applications.

Les bases de données XML natives peuvent stocker des documents XML sans en connaître le schéma, à supposer qu'il en existe un. Bien que les bases de données étendues puissent générer des schémas au vol, c'est irréalisable en pratique, surtout quand il faut traiter avec des documents ne possédant pas de schéma.

Enfin, en utilisant les NXDs, la seule interface pour accéder aux données est XML (ainsi que toutes les technologies qui lui sont liées: XPath, DOM, etc.). Les bases de données étendues, en revanche, offrent des accès directs aux données, comme à travers ODBC par exemple.

Comme mentionné plus haut, Arafox articule ses travaux autour du logiciel libre. C'est pourquoi la NXD utilisée dans le Data Manager est une solution Open Source: eXist. Nous l'avons vu, il existe également deux autres solutions libres que sont dbXML et Xindice. Les sections suivantes ont pour but de donner un aperçu de ces solutions.

III.1.2 – dbXML

dbXML est une base de données XML native, développée par le groupe dbXML. Elle supporte quatre banques de données différentes:

- Une banque de données propriétaire utilisant des arbres balancés.
- Une banque de données volatile utilisée pour le stockage temporaire et dont le contenu est supprimé lorsque la base de données est arrêtée.
- Le système de fichiers.
- Un mappage à une base de donnée relationnelle.

dbXML possède un modèle de collections. Celles-ci peuvent être organisées en hiérarchies et peuvent contenir des documents correspondant à n'importe quel schéma XML. Toutefois, afin de simplifier l'indexation et les requêtes, il est conseillé que les documents d'une seule collection correspondent à un schéma XML unique. Une collection peut contenir des flux binaires, mais elle ne stockera jamais en même temps des documents XML.
dbXML supporte XPath, XSLT, XUpdate et la recherche full-text. Elle supporte également les trois types d'index suivants:

- Les index de noms qui indexent les noms d'éléments et d'attributs.
- Les index de valeurs qui indexent les valeurs d'éléments et d'attributs. Les valeurs supportées sont les chaînes de caractères, les caractères, les octets, les nombres entiers, les nombres réels et les booléens.
- Les index full-text qui indexent les symboles dans les valeurs d'éléments et d'attributs. Ils sont insensibles à la casse et indexent la racine des mots.

Des index individuels sont associés à une collection particulière et les utilisateurs spécifient ce qu'il faut indexer selon une expression de type XPath.
dbXML supporte les triggers à travers des classes Java spécifiées par l'utilisateur. Les triggers peuvent être utilisés, par exemple, pour valider des documents lors d'une insertion, ou modifier des documents lors de la récupération. D'autres classes Java permettent également des extensions au serveur.
Les problèmes de transactions et de sécurité sont supportés par dbXML. Par défaut, le niveau de sécurité d'application est lié à la fonction de l'utilisateur. Mais il est également possible de fixer le niveau de sécurité à zéro (aucune sécurité) ou de limiter l'accès à un seul utilisateur.

Les APIs offertes par dbXML sont les suivantes:

- L'API directe qui permet aux applications de travailler directement avec dbXML
- L'API cliente qui permet aux applications d'utiliser dbXML en mode client-serveur
- L'API XML:DB
- L'interface des services Web qui supporte XML-RPC et REST

Enfin, dbXML offre une série d'outils de lignes de commande pour se connecter à la base de données, stocker des documents, les récupérer et gérer les collections, les index, la sécurité, les triggers ainsi que les extensions.

III.1.3 – eXist

eXist ([MEI02) est une base de données native XML développée par Wolfgang Meier. Elle est écrite entièrement en Java et utilise une banque de données propriétaire (arbres balancés et fichiers paginés). Elle est construite sur un système de base de données relationnelle, MySQL ou PostgreSQL. Elle peut fonctionner en tant que serveur de base de données autonome, en tant que base de données emboîtée ou en tant que moteur de servlet d'une application Web. Les documents stockés sont organisés en une hiérarchie de collections. Une collection peut contenir des collections filles et n'impose pas un type de document ou un schéma particulier aux documents.
Le langage de requêtes proposé par eXist est XQuery. Les expressions XQuery permettent de demander n'importe quelle combinaison de collections et de documents. En réalité eXist implémente la version 1.0 de XQuery de novembre 2003, à l'exception des fonctionnalités concernant le Schéma XML[6]. Toutefois eXist fournit un certain nombre d'extensions à XQuery. En particulier, le support XQuery de eXist permet les fonctions suivantes:

- L'exécution de recherches full-text.
- Les appels à l'API XML:DB.
- L'exécution d'expressions XQuery construites de manière dynamique.
- L'application de feuilles de styles XSLT à un nœud.
- Le travail avec http.
- L'exécution de méthodes Java arbitraire.

De plus, eXist fournit un support pour XInclude et XPointer. En ce qui concerne les mises à jour, elles sont supportées principalement par l'intermédiaire de XUpdate. eXist supporte l'API XML:DB ainsi que des services additionnels pour la préparation et l'exécution des expressions XQuery, la gestion des utilisateurs, la gestion des instances multiples de bases de données et les index de requêtes. DOM et SAX sont également supportés par eXist pour les documents provenant de l'API XML:DB.
Il est par ailleurs possible d'appeler eXist via XML-RPC, une interface de services Web de type REST, SOAP et WebDAV. eXist indexe de manière automatique toute structure d'éléments et d'attributs. Par défaut, elle crée des index full-text sur toutes les valeurs d'attributs et de textes, mais l'utilisateur peut désactiver cette option pour les parties de document de son choix. eXist supporte les accès en lecture/écriture concurrents pour les utilisateurs multiples, de même que le contrôle d'accès au niveau du document ou de la collection. A l'heure actuelle, eXist ne supporte pas les transactions.

[6] Vous trouverez la documentation concernant le support XQuery de eXist sur [EXIST].

III.1.4 – Xindice

Xindice ([GAI02]), développée par Apache Software Foundation est le prolongement du projet dbXML[7]. Cette base de données XML native est écrite entièrement en Java et permet le stockage d'un grand nombre de petits documents XML. Elle peut indexer des valeurs d'éléments et d'attributs et compresser les documents afin d'économiser de la place. Ici aussi, les documents sont organisés en hiérarchie de collections. Mais, contrairement à eXist, le langage de requête supporté par Xindice est XPath.
En ce qui concerne les mises à jour, Xindice supporte également le langage XUpdate de l'Initiative XML:DB. Enfin, Xindice permet aux utilisateurs de remplacer ou insérer du contenu dans un document XML.
Les APIs supportées par Xindice sont les suivantes:

- L'API XML:DB.
- Une API CORBA.
- Un plugin XML-RPC qui supporte les accès depuis des langages comme PHP ou Perl.

Pour terminer, Xindice propose comme dbXML, une série d'outils de lignes de commande pour l'utilisation et l'administration de la base de données.

III.2 – Un moteur d'indexation et de recherche

Le Data Manager dispose d'un moteur d'indexation et de recherche, pour permettre la recherche par mot clé.
Dans cette section, nous introduirons la nécessité de posséder un moteur d'indexation et de recherche, pour un système de gestion de documents multimédia. Ensuite nous aborderons le moteur de recherche et d'indexation utilisé dans le Data Manager, qui est le projet Lucene de la fondation Apache.

III.2.1 – Pourquoi utiliser un moteur de recherche? ... et d'indexation

La nécessité de retrouver facilement et simplement divers documents, au sein des entreprises ou sur l'Internet a donné naissance au développement de moteurs de recherche.

Essayons d'imaginer ce que serait un outil de gestion de documents multimédia, comme le Data Manager sans moteur de recherche. Dans ce cas, un utilisateur désireux de retrouver une photo de Marylin Monroe, devrait parcourir chaque photo du système jusqu'à ce qu'il en trouve une correspondant à son souhait. Ainsi, chaque recherche prendrait une éternité, d'autant plus que, s'il veut être certain de trouver toutes les photographies de Marylin Monroe, l'utilisateur devrait parcourir toutes les photos du système, jusqu'à la dernière. C'est pourquoi, il est nécessaire pour le Data Manager de proposer un moteur de recherche. Ainsi, il suffira à notre utilisateur de questionner le système à l'aide d'une requête, en utilisant le mot clé "Marylin Monroe".

[7] Toutefois, la version actuelle de dbXML étant une réécriture complète du code à l'origine de Xindice, ces deux bases de données XML natives sont des produits différents. Notons que, selon [XINDICE], la fondation Apache a hérité du code de dbXML en décembre 2001.

A cette fin, lors du stockage d'un document multimédia, le Data Manager analyse sa description et la stocke dans une base de données, sous un format optimisé pour la recherche. Et c'est précisément le travail que fait un moteur d'indexation, en créant des index de chaque document en vue de faciliter leur recherche et leur récupération.

Les bases de données XML natives telles que celle utilisée dans le Data Manager offrent également des outils de recherche. Toutefois, ceux-ci ne permettent pas certaines recherches avancées, comme le boosting, que nous verrons plus loin. C'est pourquoi, le Data Manager utilise Lucene, un projet Open Source du groupe Jakarta Apache.

III.2.1 – Lucene

Lucene fait partie intégrante du projet Jakarta Apache (*Apache Jakarta Project*). Ce projet a pour but de d'offrir des applications serveur de qualité commerciales, et ce dans un esprit ouvert et de coopération.
Le groupe Jakarta Apache décrit Lucene comme étant "*une librairie de moteur de recherche texte complet, entièrement écrite en Java*" (voir [JAKARTA]). Il s'agit d'une technologie convenant à presque toutes les applications nécessitant de la recherche full-text. Plus précisément, Lucene est une API permettant de développer et personnaliser son propre moteur d'indexation et de recherche. [LIN03] décrit comment intégrer un moteur de recherche à un site Web.

Selon [TAN04], la qualité d'un moteur de recherche tient dans sa rapidité à répondre à une requête, et une condition essentielle pour fournir des réponses rapidement est d'avoir un bon format de stockage pour les index.

A – Les formats de fichiers d'index

Comme le montre la figure III.4, Lucene possède quatre éléments fondamentaux:

- L'**index** qui contient une série de documents.
- Le **document** qui est une suite de champs.
- Le **champ** qui est une suite de mots.
- Le **mot** qui est une chaîne de caractères.

En réalité, l'occurrence d'un mot est représentée par un couple (*field*, *text*), où *field* est une chaîne de caractères représentant le champ dans lequel il se trouve et *text* est une chaîne de caractères dénommant un texte dans ce champ. De cette façon, la même chaîne de caractères se trouvant dans deux champs différents sera considérée comme deux mots différents. Par exemple, si le titre d'un document contient une occurrence du mot "chat" et que le texte contient une autre occurrence du mot "chat", la première sera représentée par le couple (titre, chat) et la seconde sera représentée par le couple (texte, chat).

L'index de Lucene fait partie de la famille des index inversés. En effet, il peut lister, pour un mot donné, les documents qui contiennent ce mot. La relation naturelle va en sens inverse: un document liste des mots.

Dans Lucene, les champs peuvent être stockés, auquel cas leur texte est littéralement stocké dans l'index, de façon non inversée. Les champs qui sont inversés sont dits indexés. Notons qu'un champ peut être en même temps stocké et indexé.

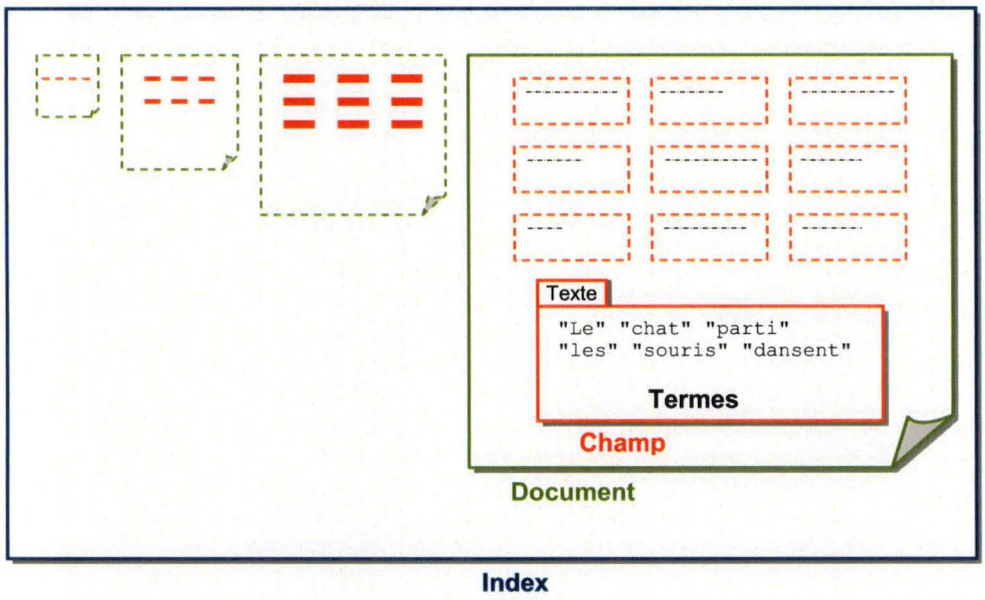


Figure III.4. Les éléments fondamentaux de Lucene.

Pour être indexé, le texte d'un champ peut être soit *tokenizé* en mots, soit utilisé littéralement comme un mot. La plupart des champs sont *tokenizés*, mais il est parfois utile pour certains champs identifiants d'être indexés littéralement.

Les index peuvent être composés de plusieurs segments. Un segment est un sous-index, mais est considéré comme un index totalement indépendant. L'évolution des index se fait par création de nouveaux segments pour les documents ajoutés dernièrement et par fusion des segments existants. Les recherches peuvent concerner plusieurs segments et/ou plusieurs index, chaque index pouvant être composé d'une série de segments.

En interne, Lucene se réfère aux documents à l'aide d'un numéro de document. Ainsi, le premier document ajouté à l'index porte le numéro zéro, le deuxième porte le numéro un, etc.
Notons qu'un numéro de document peut changer, donc il faut faire attention lorsque l'on stocke ces numéros en dehors de Lucene. En effet, les numéros stockés dans chaque segment sont uniques à l'intérieur de ce segment. Donc, ils doivent être convertis avant de pouvoir être utilisés dans un autre contexte. La technique habituelle consiste à allouer à chaque segment un intervalle de valeurs, en fonction des nombres utilisés dans ce segment. Pour convertir le numéro d'un document d'un segment en une valeur externe, on y ajoute la valeur de base du segment (i.e. la première valeur de l'intervalle). Pour reconvertir une valeur externe en une valeur spécifique à un segment, le segment est identifié par l'intervalle dans lequel la valeur se trouve, et on en soustrait la valeur de base de ce segment.
Par exemple, nous pourrions imaginer deux segments comportant 5 documents, comme l'illustre la figure III.5. Ces segments peuvent être combinés de sorte que le premier ait une base de 0 (i.e. un intervalle de valeurs [0, 4]) et le second ait une base de 5 (i.e. un intervalle de valeurs [5, 9]). Ainsi, le document numéro 3 du premier segment aurait une valeur externe de 3 alors que le document numéro 3 du second segment aurait une valeur externe de 8 (cf. figure III.5 (a)). De la même façon, un document ayant pour valeur externe 5, c'est-à-dire une valeur se trouvant dans l'intervalle [5, 9], correspond au document numéro 0 du deuxième segment (cf. figure III.5 (b)).
En outre, lorsque les documents sont supprimés, cela crée des trous dans la numérotation. Ceux-ci peuvent être supprimés, lors de la fusion. En effet, lorsque les segments sont

fusionnés, les documents supprimés sont éliminés. Un segment qui vient d'être fusionné n'a donc pas de trou dans sa numérotation.

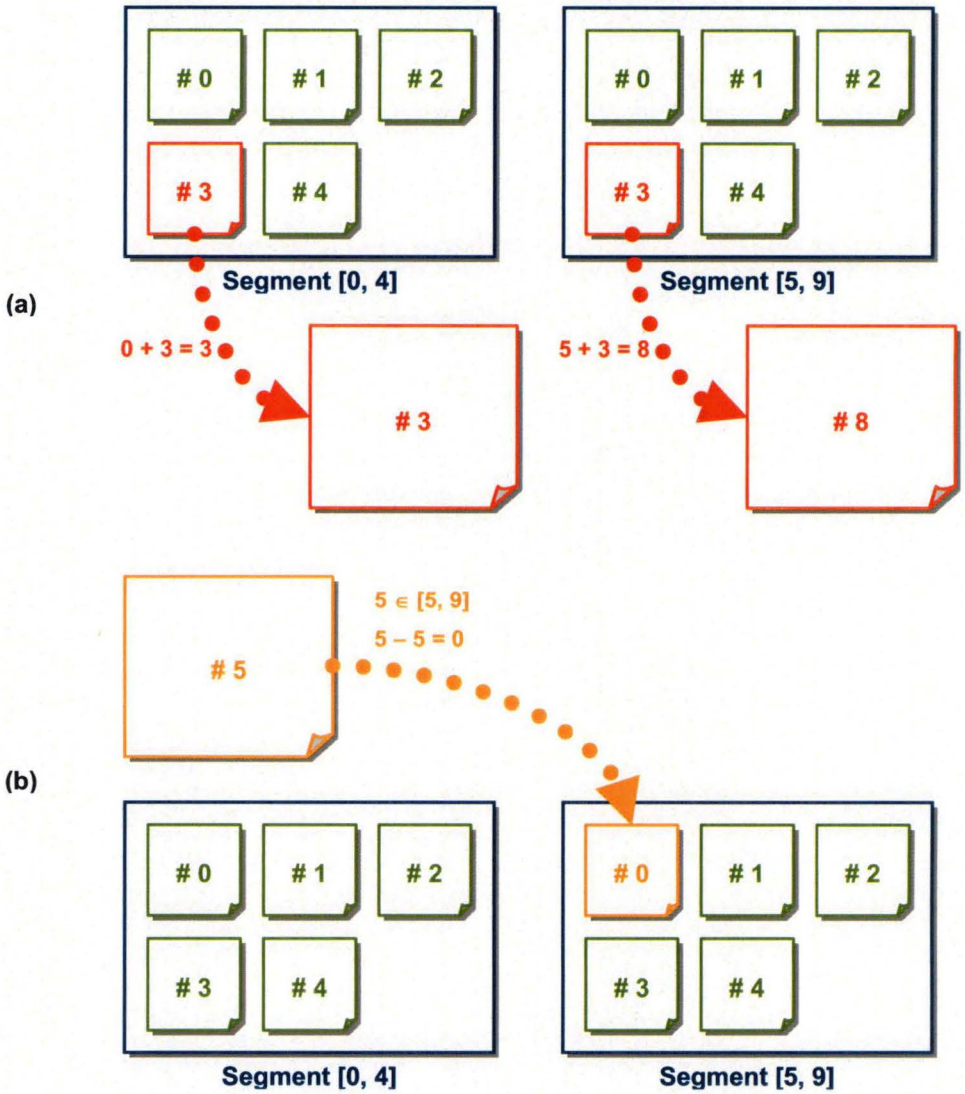


Figure III.5. La numérotation des documents dans Lucene. **(a)** Calcul de la valeur externe d'un numéro de document. **(b)** Calcul de la valeur interne d'un numéro de document.

Chaque index de segment maintient notamment les éléments suivants:

- Les **noms de champ** (*field names*) qui contiennent l'ensemble des noms de champ utilisés dans l'index.
- Les **valeurs de champs stockés** (*stored field values*) qui contiennent, pour chaque document, une liste de couples (attribut, valeur), où les attributs sont des noms de champ. Les valeurs de champs stockés sont utilisées pour stocker de l'information supplémentaire sur le document (par exemple le titre, l'url, ou l'identifiant pour accéder à la table). L'ensemble des champs stockés correspond à ce qui est renvoyé pour chaque hit, lorsque l'on effectue une recherche.
- Le **dictionnaire de mots** (*term dictionary*) qui contient tous les mots utilisés dans tous les champs indexés de tous les documents. Il contient également le nombre

de documents comprenant le mot et des pointeurs vers les fréquences et proximités de mot.

- La **fréquence de mot** (*term frequency data*) donne, pour chaque mot du dictionnaire, le nombre de documents qui contiennent ce mot et la fréquence de ce mot dans le document.
- La **proximité de mot** (*term frequency data*) donne, pour chaque mot du dictionnaire les positions où il apparaît dans chaque document.
- Les **documents effacés** (*deleted documents*) constituent un fichier optionnel indiquant les documents supprimés.

Un exemple concret d'indexation avec Lucene

Lorsque l'on veut indexer un document avec Lucene, il faut créer, pour celui-ci, un document Lucene. Cela se fait en ajoutant des champs correspondant aux critères sur lesquels pourront se baser les recherches ultérieures.
Imaginons, par exemple, que l'on désire concevoir un moteur de recherche d'articles de journal, de façon à pouvoir les récupérer sur base du titre, de la date de parution, de l'auteur ou encore du thème (i.e. des termes contenus dans le corps de l'article). Comme nous pouvons le voir à la figure III.6, il faudrait dans ce cas créer un document Lucene pour chacun des articles, en ajoutant, par exemple:

- Un champ *Titre* contenant le titre.
- Un champ *Date* contenant la date de parution.
- Un champ *Auteur* contenant le nom de l'auteur ou de l'agence de presse.
- Un champ *Texte* contenant le corps de l'article.

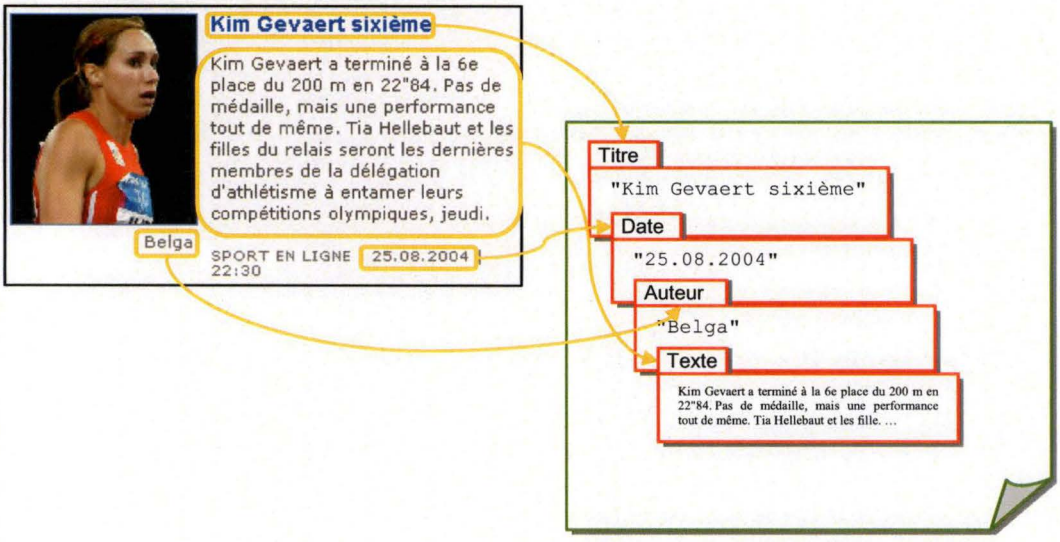


Figure III.6. Création d'un document Lucene.

Une fois que le document Lucene est créé, celui-ci peut faire l'objet d'une indexation sur base de ces quatre champs. Donc, un fichier texte, par exemple, ne peut pas être indexé tel quel dans Lucene, mais son contenu doit être d'abord représenté dans un format propre à Lucene.

L'appellation des fichiers

Tous les fichiers d'un même segment on le même nom, à l'extension près. Celles-ci correspondent aux différents formats de fichiers décrits ci-dessous.

En général, tous les segments d'un index sont stockés dans un même répertoire, mais cela n'est pas obligatoire.

Les types primitifs

Les types primitifs utilisés par Lucene pour coder ses données sont l'octet, l'UInt32, l'UInt64, le VInt, le caractère et la chaîne de caractères.

Les fichiers par index

Les segments actifs d'un index sont stockés sans un fichier *segment info file*. Chaque index n'a qu'un seul fichier de ce format, et est nommé "*segments*". Ce fichier liste tous les segments par leur nom et contient la taille de chaque segment.

Lucene utilise plusieurs fichiers pour indiquer qu'un autre processus utilise un index. Ils ne sont pas stockés dans le même répertoire que l'index, mais dans le répertoire temporaire.
La présence d'un certain fichier ("*commit.lock*") indique soit qu'un processus est en train d'écrire dans le fichier "*segments*" et d'effacer les fichiers d'index périmés du segment, soit qu'un processus est en train de lire le fichier "*segments*" et d'ouvrir les fichiers des segments qu'il nomme. Ce fichier lock empêche les fichiers d'être supprimés par un autre processus après qu'un processus ait lu le fichier "*segments*", mais avant qu'il ait tenté d'ouvrir tous les fichiers des segments qu'il nomme.
La présence d'un autre fichier ("*write.lock*") indique q'un processus est en train d'ajouter des documents dans un index, ou de retirer des fichiers de cet index. Ce fichier lock empêche plusieurs processus d'essayer de modifier un index au même moment.

Un fichier de destruction, appelé "*deletable*" contient les noms des fichiers qui n'on plus été utilisés par l'index, mais qui n'ont pas encore été supprimés.

Les fichiers par segment

Les noms de fichiers sont stockés dans le fichier "*field info*", avec le suffixe .fnn.
Les champs sont numérotés par leur ordre dans ce fichier. Donc, le champ 0 est le premier champ dans le fichier, le champ 1 est le suivant, etc. Comme pour les numéros de documents, les numéros de fichiers sont relatifs au segment.

Les champs de stockage sont représentés par deux fichiers: "*field index*" et "*field data*". Le premier fichier, appelé aussi .fdx, contient pour chaque document un pointeur vers sa donnée de champ. Il est utilisé pour trouver la position dans le fichier "*field data*" du champ d'un document donné. Le second fichier, appelé aussi .fdt, contient les champs stockés pour chaque document.

Un dictionnaire de mots est représenté par deux fichiers que sont les fichiers "*term infos*" (ou .tis) et "*term info index*" (ou .tii). Le premier fichier est trié par mot. Les mots sont d'abord classés par ordre alphabétique en fonction du nom de champ du mot. Ensuite, à l'intérieur par ordre alphabétique sur le texte du mot. Pour économiser de la place, Lucene

note le préfixe que le mot a en commun avec le mot qui le précède, suivi de la chaîne de caractères qui lui est propre. De cette façon, si le mot "fillette" est précédé du mot "fille", il sera codé "4 ette". Le fichier .tii contient de l'information concernant le fichier .tis.

Lucene utilise également d'autres fichiers d'information:

- Le fichier "*frequencies*" (ou .frq) reprenant les listes des documents qui contiennent chaque terme, ainsi que la fréquence de ce terme dans ce document.
- Le fichier "*positions*" (ou .prx) qui contient les listes des positions où le mot apparaît dans les documents.
- Le fichier optionnel "*deleted documents*" (ou .del) qui indique si un segment contient des suppressions.

Selon [TAN04], ce format de données permet de jongler entre la mémoire vive et le disque, en fonction des besoins de performance de l'utilisateur. De plus, le dictionnaire permet d'accéder rapidement aux données indexées.

L'API de Lucene permet à l'utilisateur de créer ses propres requêtes. Toutefois, Lucene propose également un parseur de requêtes (*Query Parser*) relativement riche. La section suivante a pour but d'explorer la syntaxe de ce parseur.

B – La syntaxe des requêtes Lucene

Une requête Lucene est décomposée en termes et opérateurs. Les termes sont de deux types:

- Le terme simple qui est un mot unique comme "chat" et "souris".
- La phrase qui est un groupe de mots entre guillemets comme "le chat parti les souris dansent".

Pour la création de requêtes plus complexes, Lucene permet de combiner plusieurs termes ensemble à l'aide d'opérateurs booléens.

Lorsqu'il réalise une recherche, l'utilisateur peut soit spécifier un champ, soit utiliser le champ par défaut. Il a la possibilité de rechercher n'importe quel champ, en introduisant son nom suivi du symbole ":" et le terme qu'il recherche pour ce champ.
Imaginons, par exemple, qu'un index Lucene contienne les deux champs *Texte* et *Titre*, et que ce dernier soit le champ par défaut. Cette situation est illustré à la figure III.7 (a). Si nous voulions retrouver le document intitulé "Proverbe" et contenant le texte "le chat parti, les souris dansent", nous pourrions effectuer la requête illustrée à la figure III.7 (b). Etant donné que le champ par défaut est *Titre*, l'indicateur de champ n'est pas nécessaire. C'est pourquoi, la requête illustrée à la figure III.7 (c) est également valide. En revanche, étant donné qu'un champ n'est valide que pour le terme qui le suit, la requête illustrée à la figure III.7 (d) ne correspond pas à ce que l'on recherche. En effet, elle trouverait "Le" dans le champ *Texte* et les termes "chat", "parti", "les", "souris" et "dansent" dans le champ *Titre* (i.e. le champ par défaut).

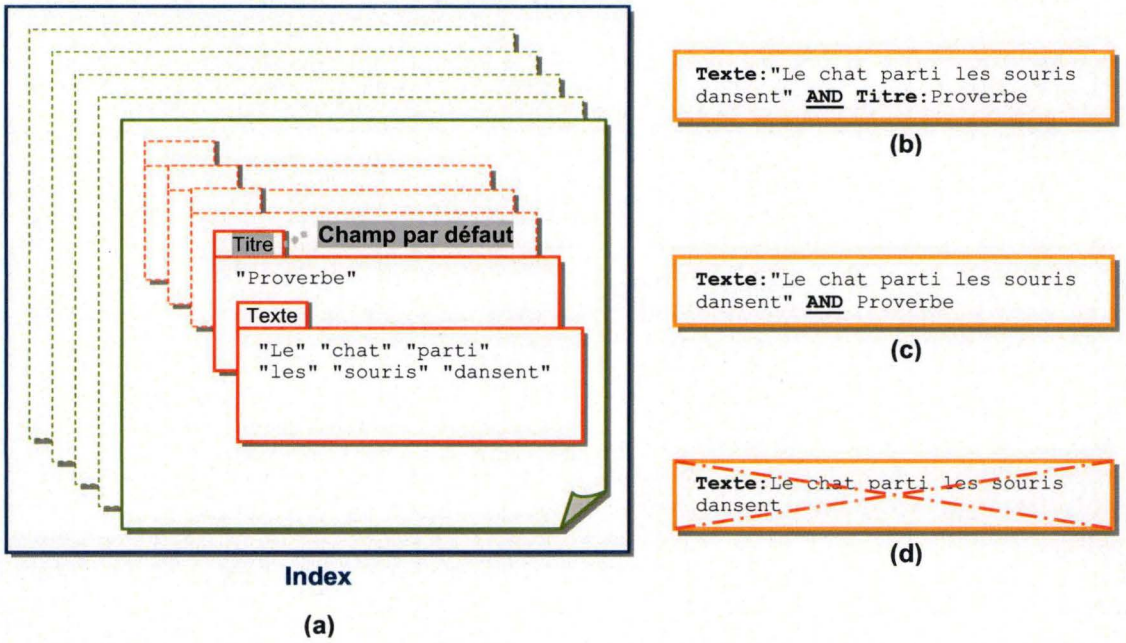


Figure III.7. Requêtes Lucene. **(a)** Index contenant les champs Texte et Titre avec le champ Titre par défaut. **(b)** et **(c)** Deux requêtes Lucene valides. **(d)** Une requête Lucene non valide.

Comme nous pouvons le voir à la figure III.8, Lucene propose un large éventail d'options de recherche à l'aide de modificateurs de termes:

- La recherche joker (*wildcard search*).
- La recherche floue (*fuzzy search*).
- La recherche de proximité (*proximity search*).
- La recherche d'intervalle (*range search*).
- Le grossissement d'un terme (*boosting a term*).

La recherche joker proposée par Lucene s'applique à un seul caractère, en utilisant le symbole "**?**" ou plusieurs, avec le symbole "*****".
La recherche joker à un seul caractère cherche les termes qui peuvent s'appliquer au caractère remplacé. Par exemple, en utilisant la requête de la figure III.8 (a), nous pourrions retrouver les documents contenant les mots "porte" ou "poste".
La recherche joker à caractères multiples cherche après 0 ou plusieurs caractères. Par exemple, en grâce à la requête de la figure III.8 (b), nous pourrions retrouver les documents contenant les mots commençant par le préfixe "port" (i.e. "port", "porte", "porter", etc.). De même qu'en utilisant la requête de la figure III.8 (c), nous pourrions retrouver les documents contenant les mots "frêle", "frère", "frégate", "fresque", "fréquente", etc.
La recherche joker de Lucene n'autorise pas l'utilisation des symboles "**?**" et "*****" à la place des premiers caractères d'un terme.

La recherche floue de Lucene s'effectue en utilisant le symbole "**~**" à la fin d'un terme simple. Donc, pour réaliser une recherche concernant les mots ayant une orthographe proche de celle du mot "futée", on utilisera la requête de la figure III.8 (d). On retrouverait ainsi les documents contenant les termes "futée, "fusée", "fumée" ou "futées".

La recherche de proximité réalisée par Lucene consiste à rechercher les documents contenant les mots d'une phrase donnée, séparés par une certaine distance. Pour cela, on utilise le symbole "**~**" à la fin d'une phrase, suivi de la distance désirée. Par exemple, pour

chercher les documents contenant les mots "chat" et "souris" séparés l'un de l'autre par deux mots, on utilisera la requête de la figure III.8 (e).

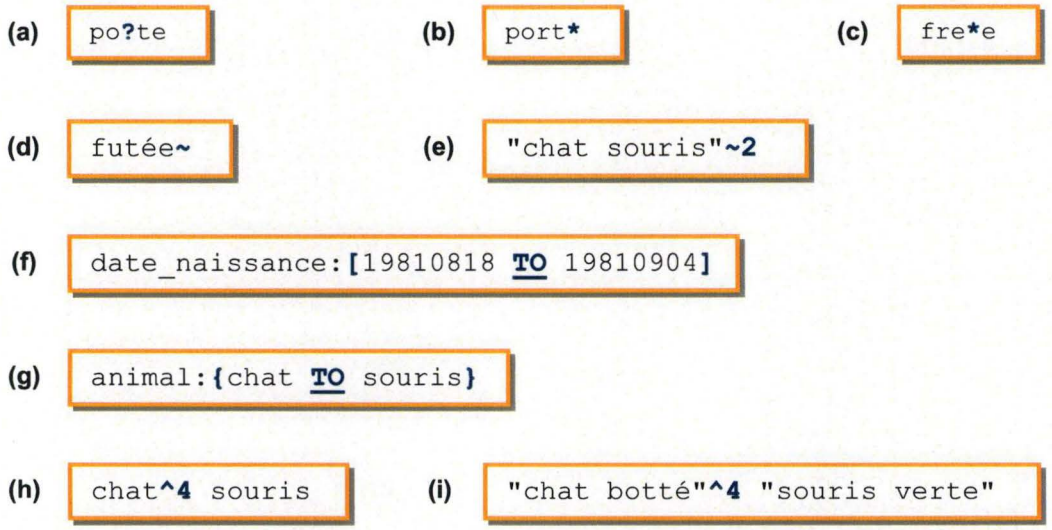


Figure III.8. Les options de recherche Lucene. **(a)** La recherche joker à caractère unique. **(b)** et **(c)** La recherche joker à caractères multiples. **(d)** La recherche floue. **(e)** La recherche de proximité. **(f)** et **(g)** La recherche d'intervalle. **(h)** et **(i)** Le boosting.

La recherche d'intervalle proposé par Lucene permet de trouver les documents dont les valeurs de champs sont entre la limite inférieure et la limite supérieure d'un intervalle donné. Ces deux limites peuvent être soit inclues dans la recherche (en utilisant les symboles "[" et "]"), soit exclues (à l'aide des symboles "{" et "}") de celle-ci. A titre d'exemple, la requête de la figure III.8 (f) cherche les documents contenant une date de naissance comprise entre le 18 août 1981 et le 4 septembre 1981 ou l'une de ces deux dates. En revanche, la figure III.8 (g) illustre une requête qui donnera pour résultat tous les documents qui concernent un animal se trouvant strictement entre le "chat" et la "souris", par ordre alphabétique.

Lucene permet de préciser le niveau de pertinence d'un terme particulier. Il est donc possible de "booster" un terme, c'est-à-dire de lui donner de l'importance, en utilisant le symbole "^" à la fin, suivi d'un facteur de grossissement. Par défaut, ce facteur vaut 1, mais il peut prendre n'importe quelle valeur positive, même inférieure à 1. Plus le facteur de grossissement sera grand, plus le terme aura de l'importance dans la requête. Cette méthode permet de contrôler la pertinence d'un document, en grossissant ses termes.
A titre d'exemple, supposons que nous recherchions les documents concernant un chat et une souris, et que nous voulions que le terme "chat" soit plus significatif. Nous pourrions "booster" ce terme, en utilisant une requête similaire à celle qui est illustrée à la figure III.8 (h). De cette façon, nous donnerions plus d'importance à la présence du mot chat dans le document qu'à celle du mot souris. Ici, le facteur de grossissement vaut 4.
Lucene permet également de grossir une phrase. En effet, comme illustré à la figure III.8 (i), il est possible de donner plus d'importance à la présence de la phrase "chat botté" qu'à celle de la phrase "souris verte". Ici encore, le facteur de grossissement vaut 4.

En plus de ces options de recherche, Lucene permet la combinaison de termes à au moyen d'opérateurs logiques. Les opérateurs booléens supportés sont **AND**, "+", **OR**, **NOT** et "-", et sont illustré à la figure III.9.

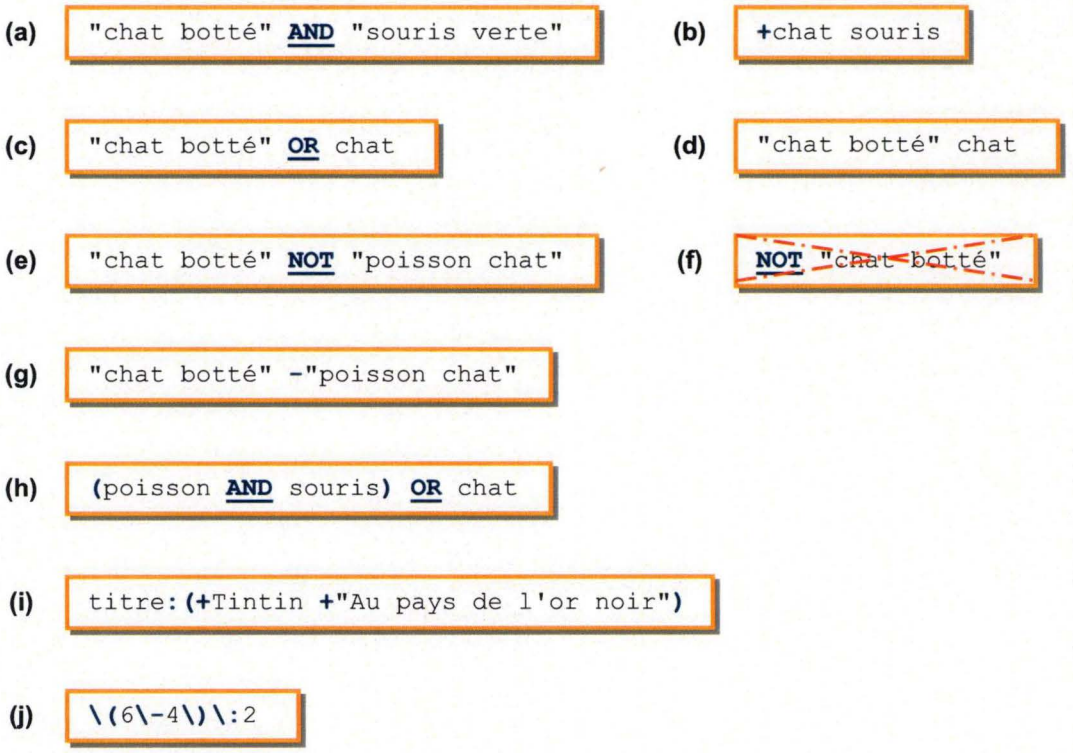


Figure III.9. Les opérateurs logiques de Lucene. **(a)** L'opérateur **AND**. **(b)** L'opérateur d'exigence +. **(c)** et **(d)** L'opérateur **OR**. **(e)** et **(f)** L'opérateur **NOT**. **(g)** L'opérateur d'interdiction -. **(h)** Le groupement de clauses. **(i)** Le groupement de champs. **(j)** L'utilisation des caractères réservés.

L'opérateur **AND** (ou **&&**) permet de retrouver les documents contenant les deux termes qu'il lie. C'est l'équivalent d'une intersection pour les ensembles. A titre d'exemple, la requête de la figure III.9 (a) permet de retrouver les documents contenant "chat botté" et "souris verte".

L'opérateur "+" de Lucene indique que le terme qui le suit est requis, c'est-à-dire qu'il doit exister quelque part, dans le champ d'un document. Par exemple pour rechercher les documents qui doivent contenir le terme "chat" et peuvent contenir le terme "souris", on peut utiliser la requête de la figure III.9 (b).

L'opérateur **OR** (ou **||**) est l'équivalent de l'union pour les ensembles. Il lie deux termes entre eux et permet de trouver un document répondant à la requête si celui-ci contient l'un des termes en question. **OR** est l'opérateur de conjonction par défaut. Donc, s'il n'y a aucun opérateur booléen séparant deux termes, Lucene utilise **OR**. Par exemple, les requêtes de la figure III.9 (c) et (d) permettent toutes les deux de retrouver les documents contenant soit "chat souris", soit "chat".

L'opérateur **NOT** (ou **!**) permet d'exclure de la recherche tous les documents contenant le terme qu'il précède. Il s'agit de l'équivalent de la différence pour les ensembles. A titre d'exemple, la figure III.9 (e) illustre une requête permettant de retrouver les documents qui contiennent "chat botté", mais pas "poisson chat". En revanche, la figure III.9 (f) illustre une requête non valide. En effet l'opérateur **NOT** de Lucene ne peut pas être utilisé avec un seul terme ou une seule phrase.

L'opérateur d'interdiction ou "-" permet d'exclure des documents qui contiennent le terme qu'il précède. Par exemple, pour rechercher les documents qui contiennent "chat botté", mais pas "poisson chat", on peut utiliser la requête illustrée à la figure III.9 (g).

Lucene permet de grouper des clauses pour former des sous-requêtes. A cette fin, on utilise les symboles "(" et ")". A titre d'exemple, pour rechercher les documents contenant soit les termes "poisson" et "chat", soit les termes "souris et chat", nous pouvons utiliser la requête de la figure III.9 (h). Cette technique permet d'éviter la confusion et de s'assurer que le document contient le terme "chat" et qu'il peut également contenir les termes "poisson" et "souris".
Les parenthèses permettent également de grouper plusieurs clauses dans un seul champ. Ainsi, par exemple, pour rechercher les documents dont le titre contient le mot "Tintin" et la phrase "Au pays de l'or noir", nous pourrions utiliser la requête illustrée à la figure III.9 (i).

Enfin, Lucene fournit un mécanisme pour utiliser les caractères réservés du parseur. Ces caractères spéciaux sont "+", "-", "&&", "||", "!", "(", ")", "{", "}", "[", "]", "^", """, "~", "*", "?", ":", et "\". Pour éviter la prise en compte de ces caractères, il est possible de les préfixer du symbole "\". Ainsi, par exemple, pour rechercher les documents contenant le terme "(6-4):2", nous pouvons utiliser la requête de la figure III.9 (j).

Avant de choisir d'utiliser le parseur de requêtes de Lucene il faut prendre en considération que celui-ci est destiné à du texte introduit par l'utilisateur final et non à des requêtes générées automatiquement. Par conséquent, il faut prendre cela en compte dans le cas où les requêtes sont générées par un programme en utilisant l'API de requêtes de Lucene.

C – L'architecture de Lucene

La figure III.10, issue de [TAN04] illustre la façon dont Lucene est organisée. Tout en bas, on trouve les objets d'accès aux données qui sont généralement accessibles par les classes du package `store`. Ensuite, on trouve une couche métier permettant d'accéder aux différents fichiers d'index. L'index n'est pas accessible directement par l'utilisateur, il faut passer par système d'indexation et de recherche. On trouve également des couches propres à l'"indexation et à la recherche. Etant donné que l'indexation et la recherche sont deux tâches bien distinctes, on peut séparer leurs couches respectives. En ce qui concerne les couches propres à la recherche, on trouve le parseur de requêtes et la recherche proprement dite. Les couches propres à l'indexation, quant à elles, concernent la création de documents et l'analyse de ceux-ci.

L'API de Lucene[8] est donc composée de sept packages principaux:

- **`org.apache.lucene.util`** qui contient quelques structures de données utiles telles qu'une implémentation optimisée de vecteurs de bits (`BitVector`) ou une méthode pour manipuler des chaînes de caractères (`StringHelper`).
- **`org.apache.lucene.store`** qui définit une API pour les entrées/sorties binaires. Notons que dans le but de pouvoir accéder aux données sans se préoccuper de leur support, Lucene n'accède pas directement à ce package.
- **`org.apache.lucene.document`** qui fournit une classe simple pour un `Document` (i.e. l'unité d'indexation et de recherche de Lucene). Un Document est

[8] Vous trouverez l'API finale de la version 1.4 de Lucene à l'adresse http://jakarta.apache.org/lucene/docs/api/ (accédé le 15/08/2004)

simplement un ensemble de champs nommés (`Field`) dont les valeurs peuvent être des chaînes de caractères ou des instances de `java.io.reader`.

- **`org.apache.lucene.analysis`** qui définit une API permettant de convertir du texte en objets pouvant être indexés.
- **`org.apache.lucene.index`** qui permet de maintenir et accéder aux index. Il fournit deux classes primaires: `IndexWriter` qui crée et ajoute des documents dans les indices et `IndexReader` qui accède aux données dans l'index.
- **`org.apache.lucene.search`** qui permet de rechercher dans les index. Il fournit des structures pour représenter les requêtes (i.e. `TermQuery` pour les mots uniques, `PhraseQuery` pour les phrases et `BooleanQuery` pour les combinaisons booléennes de requêtes) ainsi qu'un moteur de recherche abstrait (`Searcher`) qui retourne les résultats sous forme de `Hits` (i.e. des listes rangées de documents). Un `IndexSearcher` utilise la recherche au-dessus d'un simple `IndexReader`.
- **`org.apache.lucene.queryParser`** qui définit un simple parseur de requêtes. Les requêtes parsées pourront être utilisées par le `Searcher`.

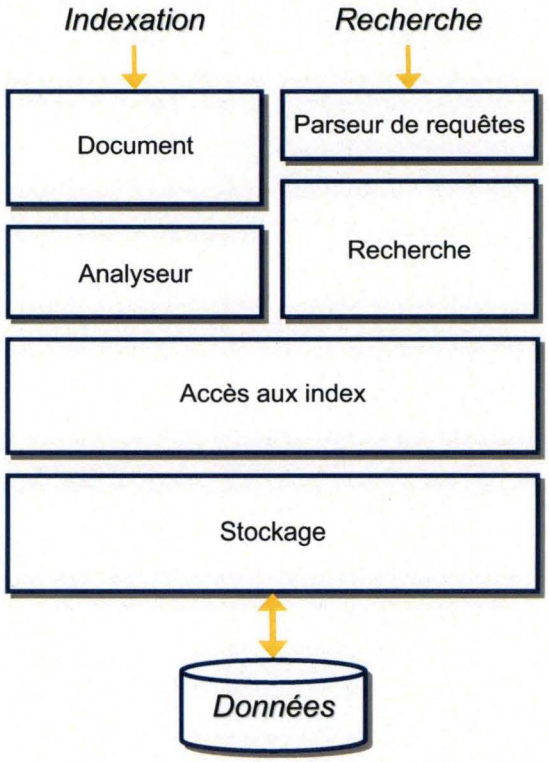


Figure III.10. Architecture de Lucene.

Pour utiliser Lucene, une application doit réaliser les étapes suivantes:

1. Créer un document (`Document`) en ajoutant des champs (`Field`).
2. Créer un `IndexWriter` et y ajouter les documents à l'aide de la méthode `addDocuments()`.
3. Appeler la méthode `QueryParser.parse()` pour construire une requête à partir d'une chaîne de caractères.
4. Créer un `IndexSearcher` et passer la requête dans la méthode `search()` de celui-ci.

Nous l'avons vu, le système d'indexation de Lucene se base sur la création de documents composés de champs. Or, dans le cas du Data Manager, les documents qu'il faut indexer sont des fichiers XML. C'est pourquoi la phase d'indexation du Data Manager doit utiliser un outil permettant de parser un document XML en un document Lucene (XML → Document). De même, lorsqu'un utilisateur effectue une recherche, le Data Manager lui fournit un résultat au format XML. Le système doit donc également disposer d'un outil permettant de parser un résultat Lucene en un document XML (Hits → XML).
De tels outils ont déjà été créés et sont disponibles sur la page du projet[9].

III.3 – WebDAV

Pour le stockage des documents multimédia à proprement parler, le Data Manager dispose d'une banque de données de type Webdav.

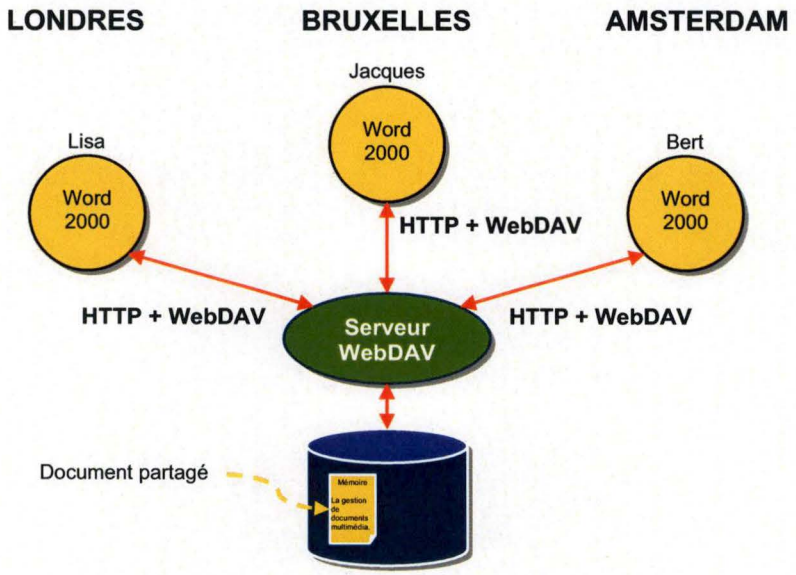


Figure III.11. Collaboration de document WebDAV.

WebDAV (pour World Wide Web Distributed Authoring and Versioning) est une extension du protocole HTTP permettant d'éditer des documents à distance ([TIS99] et [UCI99]). Plus précisément, selon [WEBDAV], il s'agit d'une "*série d'extensions au protocole HTTP qui permettent aux utilisateurs d'éditer et de gérer collectivement des fichiers sur des serveurs Web distants*". Toutefois, les opérations que WebDAV fournit font de lui plus qu'un simple outil de mise à jour Web (cf. [DEL02]). Il peut, par exemple, être vu comme un système de fichiers utilisable sur l'Internet, permettant de manipuler les fichiers un par un, tout en offrant de bonnes performances et une grande disponibilité. WebDAV peut encore être vu comme un protocole avancé qui permettrait de manipuler le contenu de documents d'un système d'information.
A titre d'exemple, la figure III.11 (inspirée de [UCI01]) montre que trois collaborateurs, situés à trois endroits différents écrivent conjointement un document, en utilisant les capacités WebDAV de Word 2000.

[9] La page concernant les convertisseurs de documents XML en documents Lucene se trouve à l'adresse http://jakarta.apache.org/lucene/docs/contributions.html.

A cette fin, WebDAV permet toute une série d'opérations. Les premières d'entre elles concernent les propriétés. On y retrouve la création, la suppression et la recherche d'informations relatives aux pages Web. Ces informations sont, par exemple, l'auteur, la date de création, etc. WebDAV permet également de lier des pages entre elles, quel que soit leur format. Il faut entendre par propriété un couple (*nom*, *valeur*) où le *nom* est une URL et la *valeur* est un document XML. L'URL permet d'ajouter de nouvelles propriétés sans devoir les enregistrer intégralement. D'autres opérations concernent les collections. Il s'agit de la création, la suppression et le listing d'un ensemble de ressources. Les collections WebDAV sont des ressources composées d'un ensemble d'URLs correspondant aux membres appelés. Un troisième ensemble d'opérations que WebDAV permet de faire concerne la prévention d'écrasement des données ou le verrouillage. Elles consistent à éviter que plus d'une personne aient accès à un document en même temps. Cela permet d'éviter que des mises à jour se perdent, ce qui arrive lorsque plusieurs utilisateurs modifient un document sans fusionner les différentes modifications entre elles. Pratiquement, WebDAV fournit un verrou exclusif en écriture et un verrou partagé. Le premier garantit que seul le propriétaire du verrou puisse écraser une ressource verrouillée. Le second, quant à lui, permet à un groupe d'utilisateurs de travailler en commun sur une ressource. D'autres opérations fournies par WebDAV concernent la gestion des espaces de noms. Il s'agit des opérations permettant de demander au serveur de copier ou de déplacer des pages Web. Ces opérations concernent des ressources uniques ou des arbres de ressources. WebDAV permet également de faire des opérations de versioning (gestion de versions): stocker les révisions importantes pour pouvoir les récupérer par après, récupérer l'historique d'une ressource ou désigner une version comme étant celle de référence. Enfin, WebDAV permet de réaliser des opérations de contrôle d'accès. Cela consiste à limiter les droits d'accès d'un utilisateur à une ressource donnée.

III.4 – La gestion de synonymes

Nous l'avons vu, le Data Manager dispose d'un moteur de recherche et d'indexation pour permettre à l'utilisateur de rechercher des documents multimédia. En ce qui concerne l'indexation, Lucene indexe les descriptions XML associées aux documents audiovisuels stockés dans le système. En réalité, les noms des éléments du document XML sont utilisés comme champs pour l'indexation alors que les valeurs des éléments sont utilisées comme termes. Donc, pour récupérer un document multimédia particulier, il faut effectuer une requête en utilisant les termes contenus dans la description de ce document.
Etant donné que le vocabulaire employé pour décrire des documents (multimédias ou autres) similaires peut différer fortement d'un individu à l'autre, il est important de prendre en considération la synonymie.

A titre d'exemple, supposons qu'un utilisateur désire retrouver toutes les photos dans lesquelles on aperçoit une voiture rouge. A cette fin, il pourrait interroger le système en lui demandant: "recherchez, s'il vous plait, toutes les photos disponibles sur lesquelles on peut voir une voiture de couleur rouge" (cf. figure III.12). Il se pourrait que l'utilisateur ne reçoive aucun résultat (ou très peu), pour la seule raison que les termes utilisés dans les descriptions pour caractériser ce type de photos soient "auto" et "pourpre". Comme on peut le voir à la figure III.12 (a), Lucene ne prend pas en compte la synonymie des différents mots de la requête. Ainsi, les résultats renvoyés par celui-ci concernent uniquement les descriptions qui comportent les mots tels qu'ils sont libellés dans la requête. Dans l'illustration, il s'agit d'un élément ayant pour nom "objet" et pour valeur "voiture" ainsi que d'un élément ayant pour nom "couleur" et pour valeur "rouge". Or il se peut que certains

documents, potentiellement intéressants soient décrits par des termes synonymes de ceux de la requête. C'est pourquoi le moteur d'indexation et de recherche du Data Manager, basé sur Lucene, prend en compte la synonymie. Grâce à cela, les résultats de la requête sont enrichis des descriptions de documents similaires, mais décrits en d'autres termes. Ceci est illustré à la figure III.12 (b).

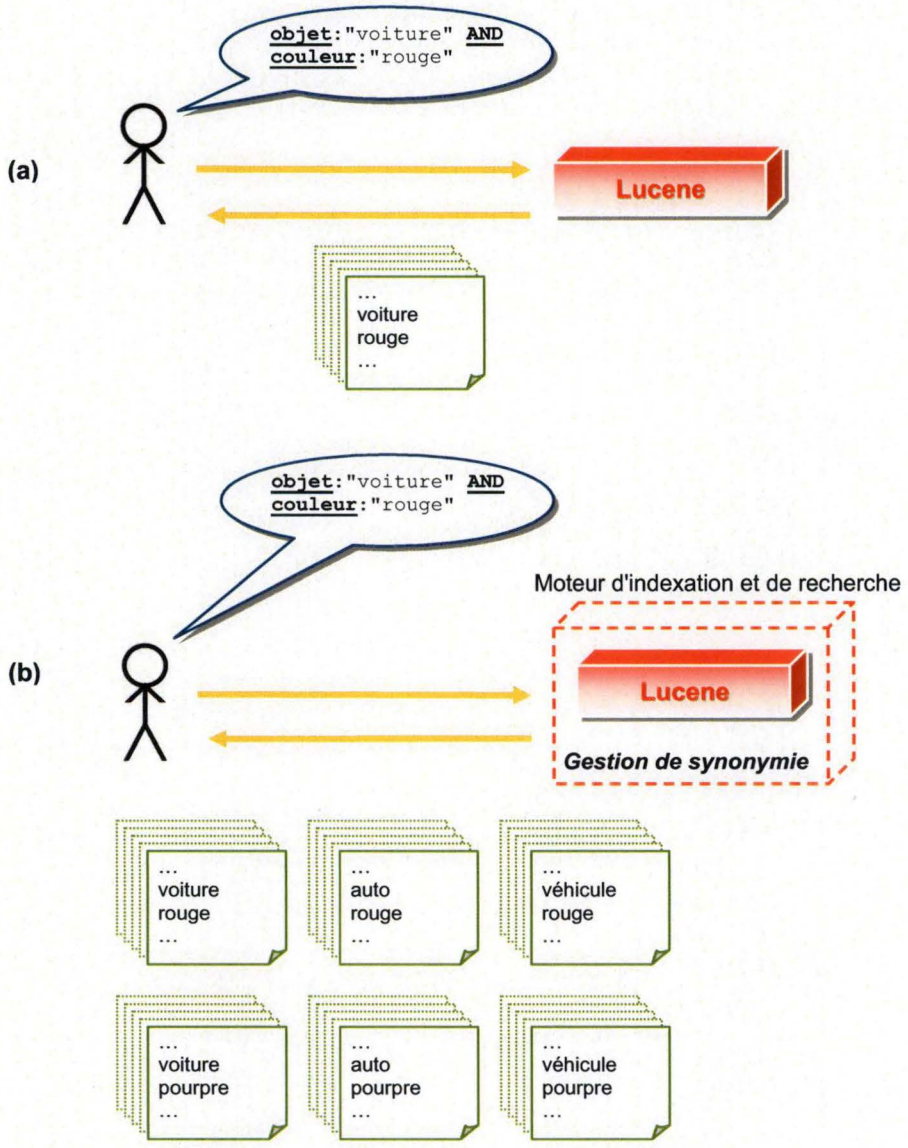


Figure III.12. Recherche de documents dans le Data Manager. **(a)** Sans prise en compte des synonymes. **(b)** Avec prise en compte des synonymes.

Lors de mon stage, chez Arafox, j'ai contribué à la conception des modules propres à la gestion de la synonymie dans le moteur de recherche du Data Manager. Les sections suivantes proposent un aperçu de ces modules.

III.4.1 – Un gestionnaire de synonymes

Le gestionnaire de synonymes du Data Manager est une API Java permettant à un utilisateur de réaliser les opérations suivantes:

- Ajouter une relation de synonymie entre deux mots.
- Retirer une relation de synonymie entre deux mots.
- Retirer un mot totalement du gestionnaire de synonymes.
- Rechercher les synonymes d'un mot.

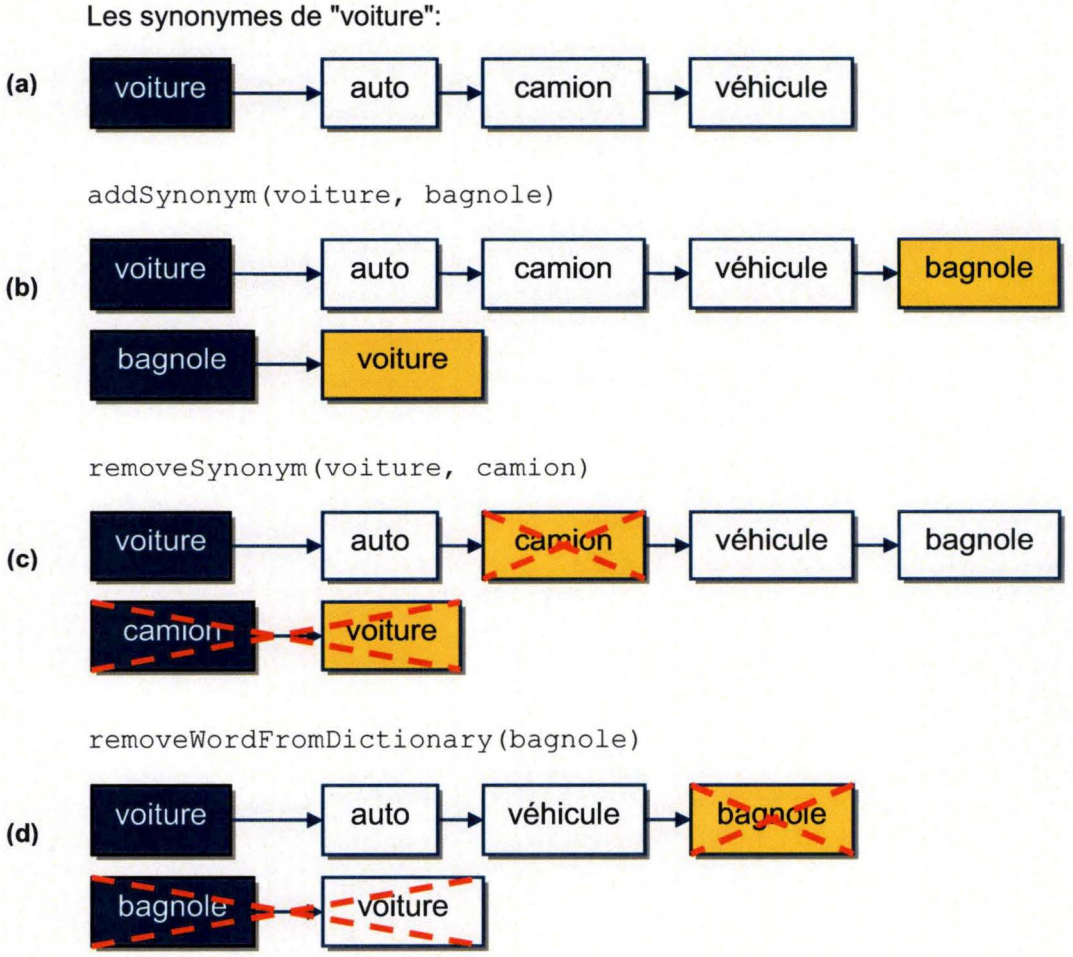


Figure III.13. La gestion des synonymes du Data Manager. **(a)** Représentation des synonymes d'un mot. **(b)** Ajout d'une relation de synonymie. **(c)** Suppression d'une relation de synonymie. **(d)** Suppression d'un mot du gestionnaire de synonymes.

Une relation de synonymie est représentée par une `SynList`, c'est-à-dire une `LinkedList` (liste chaînée Java) étendue. A titre d'exemple, la figure III.13 (a) illustre la représentation des synonymes du mot "voiture" dans le gestionnaire des synonymes. Ces listes chaînées sont stockées dans une "base de données" JDBM[10] sous une forme de table

[10] En réalité JDBM est un "*moteur de persistance transactionnel pour Java*". Il a l'avantage d'être à la fois rapide et simple, ce qui correspond aux besoins d'un gestionnaire de synonymes. Toutefois, pour faciliter la lecture, nous emploierons le terme base de données. Vous trouverez davantage d'informations sur JDBM sur la page officielle du projet: http://jdbm.sourceforge.net (accédé le 09/08/2004).

de hachage (`HashTable`). Le gestionnaire de synonymes du Data Manager travaille avec un système de cache. Cela permet de diminuer sensiblement le temps d'accès aux données.

A – Ajout d'une relation de synonymie entre deux mots

L'ajout d'une relation de synonymie entre deux mots se fait à l'aide de la méthode suivante:

```
void addSynonym(String word, String synonym)
```

Cette méthode reçoit donc un couple de mots considérés comme synonymes et ajoute la relation de synonymie entre ces deux mots. Cela consiste à ajouter chacun des mots dans la liste de synonymes de l'autre mot. Ainsi, par exemple, si l'on désire ajouter une relation de synonymie entre les mots "voiture" et "bagnole", le mot bagnole sera ajouté à la liste de synonymes du mot voiture et vice versa. Cet exemple est illustré à la figure III.13 (b). Si un mot n'a pas encore de synonyme, une nouvelle liste sera créée avec pour élément de tête ce mot. Dans l'illustration, par exemple, la liste des synonymes du mot "bagnole" a du être créée pour ajouter la relation de synonymie.

B – Suppression d'une relation de synonymie entre deux mots

La suppression d'une relation de synonymie entre deux mots se fait à l'aide de la méthode suivante:

```
void removeSynonym(String word, String synonym)
```

Cette méthode reçoit également un couple de mots considérés, cette fois, comme n'étant plus synonymes et supprime la relation de synonymie qui pourrait exister entre ces deux mots. Cela consiste à supprimer chacun des mots de la liste de synonymes de l'autre mot. Ainsi, par exemple, si l'on désire retirer la relation de synonymie entre les mots "voiture" et "camion", le mot camion sera supprimé de la liste de synonymes du mot voiture et vice versa. Cet exemple est illustré à la figure III.13 (c). Si un mot n'a plus de synonyme, sa liste de synonymes est supprimée. Dans l'illustration, par exemple, le mot "camion" n'avait plus de synonymes après la suppression de la relation de synonymie. Sa liste de synonymes a donc pu être supprimée.

C – Suppression d'un mot du gestionnaire de synonymes

La suppression d'un mot du gestionnaire de synonymes se fait à l'aide de la méthode suivante:

```
void removeWordFromDictionary(String word)
```

Cette méthode reçoit un mot qu'il faut supprimer du gestionnaire. Cela consiste à supprimer tout d'abord la liste de synonymes de ce mot, et à supprimer ensuite chaque occurrence du mot donné dans les listes de synonymes où il apparaît. Ainsi, par exemple, si l'on désire supprimer le mot "bagnole" du gestionnaire de synonymes, il faut supprimer sa liste de synonymes et son occurrence dans la liste de synonymes du mot "voiture" (puisqu'il existait une relation de synonymie entre ces deux mots). Cet exemple est illustré à la figure III.13 (d).

D – Recherche des synonymes d'un mot

La recherche de synonymes pour un mot se fait à l'aide de la méthode suivante:

```
SynList checkSynonyms(String word)
```

Cette méthode reçoit un mot dont il faut retrouver les synonymes. Cela consiste à renvoyer tout simplement la liste de synonymes du mot en question, si cette liste existe. Notons que la fonction de recherche du gestionnaire des synonymes est insensible à la casse.

Dans le cas du Data Manager, ce gestionnaire de synonymes pourrait être utilisé de deux manières différentes. Premièrement, lors de l'indexation, pour indexer les descriptions en fonction des synonymes des termes contenus dans le document XML. Deuxièmement, lors de la recherche, pour générer toutes les combinaisons possibles de requêtes, à partir des mots synonymes de ceux contenus dans la requête initiale. La première approche a l'inconvénient de surcharger inutilement les index. Cette surcharge étant proportionnelle au nombre de synonymes et de documents du système. C'est pourquoi nous avons privilégié la seconde approche qui sera développée dans la section suivante.

III.4.2 – Un générateur de requêtes

Comme introduit plus haut (cf. figure III.12 (b)), la gestion de la synonymie des différents termes d'une requête permet d'enrichir les résultats obtenus lors de la recherche. Puisque l'on dispose à présent d'un gestionnaire de synonymes, il nous est possible de développer un générateur de requêtes. Ce dernier, à partir d'une requête introduite par l'utilisateur et des synonymes des mots qu'elle contient, génère d'autres requêtes "équivalentes".

Le fonctionnement du générateur de requêtes est le suivant:

1. La requête donnée en entrée est parsée afin d'extraire les mots-clés qu'elle contient. Il s'agit de tous les mots de la requête, exception faite des mots réservés de Lucene.
2. La liste des synonymes de chaque mot-clé est constituée en interrogeant le gestionnaire de synonymes.
3. Les requêtes résultant de toutes les combinaisons possibles des synonymes sont générées.
4. Ces requêtes sont soumises au moteur de recherche Lucene.

Par exemple, la requête de la figure III.12 (b) comporte les mots-clés "voiture" et "rouge". Supposons que les synonymes de "voiture" répertoriés soient "auto" et "véhicule" et que l'unique synonyme de "rouge" soit "pourpre". Le générateur de requêtes génèrerait alors les six requêtes suivantes:

```
objet:"voiture" AND couleur:"rouge"
objet:"voiture" AND couleur:"pourpre"
objet:"auto" AND couleur:"rouge"
objet:"auto" AND couleur:"pourpre"
objet:"véhicule" AND couleur:"rouge"
objet:"véhicule" AND couleur:"pourpre"
```

Les requêtes générées sont soumises à Lucene qui fournit, dans ce cas, six séries de résultats (notons qu'une série de résultats peut être vide).

Il va de soi que toutes ces manipulations doivent être transparentes pour l'utilisateur final du Data Manager. C'est la raison pour laquelle les différentes séries de résultats doivent apparaître comme une seule et même série. En effet, l'utilisateur n'ayant introduit qu'une seule requête trouverait étonnant de recevoir plusieurs séries de résultats correspondant à d'autres mots-clés. A cette fin, un autre module du moteur de recherche se charge de regrouper les séries de résultats correspondant à une même requête initiale avant de les délivrer à l'utilisateur.

A titre d'illustration, la figure III.14 montre le fonctionnement du moteur de recherche du Data Manager. Dans un premier temps, l'utilisateur interroge le système à l'aide d'une seule requête. Ensuite, le moteur de recherche traite cette requête de la façon suivante: sur base du gestionnaire des synonymes et de la requête initiale, un premier outil (i.e. le gestionnaire de requêtes) génère différentes requêtes ayant la même signification. Celles-ci sont envoyées à Lucene qui fournira différents résultats correspondant aux différentes requêtes. A ce moment, les résultats sont fusionnés, à l'aide d'un autre module, de façon à former un unique résultat. C'est également à ce moment que le résultat est converti en un document XML, plus facile à consulter pour l'utilisateur, et renvoyé à celui-ci.

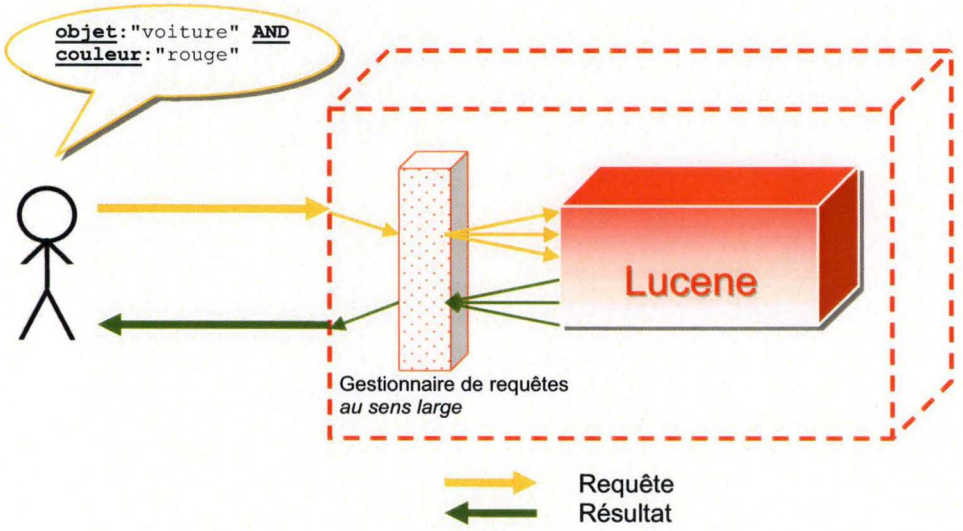


Figure III.14. Fonctionnement du moteur de recherche du Data Manager.

CHAPITRE QUATRIEME – EVALUATION DU DATA MANAGER

Ce chapitre se propose de discuter différents points de la solution du Data Manager.

IV.1 – Les bases de données XML natives et les exigences MPEG-7

Nous l'avons vu, le Data Manager traite des fichiers XML décrivant des documents multimédia. Nous avons également vu que le contenu d'un document audiovisuel pouvait être décrit à l'aide de descriptions MPEG-7. Il serait donc intéressant de pouvoir utiliser le Data Manager pour le traitement des descriptions MPEG-7. Toutefois, la gestion de descriptions MPEG-7 requiert des exigences que ne permettent pas les bases de données XML natives. Cette section, inspirée des travaux de [UTZ04], a pour but de faire un tour d'horizon de ces exigences en fonction de différentes caractéristiques spécifiques aux descriptions MPEG-7.

IV.1.1 – Le caractère non textuel des contenus des descriptions MPEG-7

Nous l'avons vu précédemment, notamment à la figure II.17, une partie non négligeable de l'information contenue dans une description MPEG-7 consiste en des données non textuelles (par exemple, l'élément contour de la même figure). Puisque les descriptions MPEG-7 sont des documents XML, ils sont par conséquent très souvent encodés comme du texte. Il en découle deux exigences fondamentales mais peu respectées dans les bases de données XML. Il s'agit de la *représentation typée* et *l'accès typé*.

Bien que cela facilite l'échange des descriptions indépendamment des plateformes, la représentation des données non-textuelles sous forme de texte n'est pas adéquate pour le stockage des descriptions dans une base de données. Non seulement les représentations textuelles requièrent plus de place, mais en plus, elles sont plus compliquées et moins efficaces à manipuler. A titre d'exemple, la représentation textuelle de la valeur de l'élément `Contour` de la figure II.17, nécessite l'utilisation d'opérations sur les chaînes de caractères pour manipuler cet élément. Ceci est évidemment plus encombrant que si on utilisait une structure de données plus appropriée aux listes (par exemple, un tableau).
Par conséquent, une solution de base de données XML native doit s'efforcer de garder les contenus de base des descriptions MPEG-7 dans une représentation typée. Cela signifie que les contenus sont encodés dans des structures de données qui conviennent à leur type particulier. Dans cette optique, une telle base de données devrait notamment supporter les types de données simples prédéfinis par le DDL MPEG-7.
En effet, une solution de base de données qui ne respecterait pas cette exigence obligerait les applications à constamment convertir les contenus non-textuels vers des représentations convenant mieux à leur traitement ultérieur. Ceci pourrait non seulement conduire à des erreurs de conversions, mais aussi engendrer des baisses de performance.

Pour les mêmes raisons, l'information non-textuelle contenue dans les descriptions MPEG-7 devrait être accessibles aux applications de la manière la plus appropriée à leur type de données particulier. Par exemple une application utilisant les valeurs de l'élément `Contour` devrait pouvoir y accéder sous la forme d'une liste de valeurs entières et non d'une chaîne de caractères.

Notons en outre que l'information textuelle contenue dans un document MPEG-7 n'est pas forcément directement interprétable par l'utilisateur. Par exemple, la signification de la valeur d'un élément `Contour` est moins intuitive qu'un chantonnement. D'où l'intérêt d'avoir recours à des méthodes d'interprétation de l'information.

Dans le cas particulier des NXDs eXist, dbXML et Xindice, ces exigences ne sont pas respectées: ni la représentation, ni l'accès aux données ne sont typés.

IV.1.2 – Le traitement partiel des descriptions MPEG-7

Une autre caractéristique des descriptions MPEG-7 est que les applications qui les manipulent ne s'intéressent généralement qu'à une partie seulement de leur contenu.
En effet, les applications manipulant des descriptions MPEG-7 utilisent seulement les parties qui sont nécessaires à l'accomplissement de leur tâche. Par exemple, un moteur de recherche de chansons par chantonnement sera principalement intéressé par les contours de la mélodie représentés par les éléments `MelodyContour` du Schéma de Description de la figure II.16. Par conséquent, il est important pour une NXD utilisée pour la gestion de descriptions MPEG-7 de stocker et de représenter la structure de telles descriptions avec une *granularité fine*. En outre, elle devrait permettre aux applications *d'accéder* au contenu des descriptions MPEG-7 avec une granularité fine.
Le prix à payer pour ces exigences est qu'un effort supplémentaire doit être fourni pour décomposer les descriptions lorsqu'elles sont importées dans la base de données, et pour les recomposer lorsqu'elles en sont exportées.
Une solution NXD ne respectant pas ces exigences forcerait les applications à charger, parser et décomposer les descriptions lors de chaque accès. Cela gênerait, en outre, la réalisation de *mises à jour fines* des descriptions.

En ce qui concernent les NXDs eXist, dbXML et Xindice, ceux-ci respectent les exigences de représentation et d'accès avec une granularité fine. L'exigence de mise à jour fine, quant à elle, n'est respectée que par dbXML et Xindice.

IV.1.3 – Exigences classiques concernant la base de données MPEG-7

Une base de donnés XML contenant des descriptions MPEG-7 est, avant tout, une base de données. Il en découle plusieurs exigences classiques pour une base de données. On s'attend à ce qu'une solution NXD offre:

- Une validation des descriptions MPEG-7, de façon à pouvoir interdire l'insertion ou la mise à jour, dans la base de données, de descriptions incohérentes avec leur Schéma de Description.
- Un support pour les transactions en garantissant les propriétés ACID (*Atomicity Consistency Isolation Durability*).
- Un contrôle fin de la concurrence, pour atteindre une concurrence élevée entre les transactions.

- Un contrôle fin d'accès aux données pour garantir et renforcer les droits d'accès des utilisateurs à des portions de données isolées.
- Des moyens fiables pour la sauvegarde (backup) et la restauration (recovery), afin d'épargner les applications, en cas de transactions avortées, de crashs systèmes, etc.

Les NXDs dbXML, eXist et Xindice, ne supportent ni la validation des descriptions, ni le contôle fin de la concurrence et de l'accès. dbXML est la seule solution à supporter les transactions. eXist, quant à lui, est le seul à offrir des moyens de backup et de recovery.

En guise de résumé, le tableau IV.1, reprend les différentes exigences MPEG-7 qui sont supportées par les trois bases de données XML natives que nous avons vues.

Tableau IV.1. Respect des exigences MPEG-7 par dbXML, eXist et Xindice.

	dbXML	eXist	Xindice
Représentation typée	-	-	-
Accès typé	-	-	-
Représentation avec granularité fine	■	■	■
Accès avec granularité fine	■	■	■
Mise à jour fine	■	-	■
Validation des descriptions	-	-	-
Transactions	■	-	-
Contrôle fin de la concurrence	-	-	-
Contrôle fin de l'accès aux données	-	-	-
Backup et recovery	-	■	-

Notons que les autres solutions NXDs et que les bases de données étendues ne permettent pas non plus la plupart des exigences demandées par les descriptions MPEG-7.

IV.1.4 – Conclusion

En conclusion, il est souhaitable qu'une solution de base de données XML destinée à gérer les descriptions MPEG-7 constitue un véritable système de gestion de bases de données au sens le plus noble du terme. De cette façon, les applications utilisatrices seraient déchargées de responsabilités sortant du cadre de leur tâche particulière.
Dès lors, pour que le Data Manager puisse être étendu à la gestion de descriptions MPEG-7, il faudra peut être réenvisager le support de base de données utilisé.

IV.2 – Lucene comme base du moteur d'indexation et de recherche du Data Manager

Nous l'avons vu à la section III.2, l'indexation et la recherche de Lucene se basent sur des formats de documents qui lui sont propres. En effet, l'indexation d'un document se fait en créant un objet `Document` et en ajoutant celui-ci dans un `IndexWriter`. En ce qui concerne la recherche, Lucene renvoie les résultats sous forme de `Hits` (i.e. une liste de documents). Dès lors, dans le cas du Data Manager, il faut passer par une phase de conversion des documents XML en documents Lucene et vice versa. Bien qu'il existe des outils permettant de réaliser cette conversion, cette phase consomme du temps et ne met pas à l'abri de perte d'informations. En effet, la structure hiérarchique d'un document XML n'est pas rendue avec le modèle de données de Lucene. C'est pourquoi, il serait peut-être plus judicieux d'utiliser un moteur d'indexation et de recherche qui traite directement les documents XML.

IV.3 – La rédaction manuelle des descriptions XML

Comme nous l'avons déjà vu, dans le Data Manager les descriptions XML associées aux documents multimédia sont créées manuellement. Ces descriptions sont donc essentiellement basées sur des caractéristiques externes au contenu (par exemple, le nom du créateur, la date de création, etc.), ou des caractéristiques subjectives (i.e. ce que la personne qui décrit le document voit – ou entend – sur celui-ci). Cela signifie que le Data Manager, tel qu'il est actuellement, est incapable de supporter d'autres requêtes que celles par mots-clés. Or, on ne peut pas tout décrire avec des mots (par exemple, la proportion exacte des couleurs dans une image, la mélodie d'une chanson, etc.). C'est pourquoi, dans le cas des images, par exemple, il serait intéressant de pouvoir effectuer des recherches ou d'approfondir les résultats en fonction du contenu du document recherché, c'est-à-dire en fonction de ses caractéristiques visuelles. De cette façon, il serait possible à un utilisateur d'effectuer des requêtes par la couleur (en précisant les couleurs qu'il désire retrouver dans l'image qu'il recherche, dans des proportions données), par la forme (i.e. les objets contenus dans l'image), etc. Il lui serait également possible d'indiquer ou de fournir une image particulière et de demander les images similaires. Dans ce cas, il faudrait, bien entendu, disposer d'un mécanisme permettant d'extraire automatiquement ce type de caractéristiques. La troisième partie de ce mémoire a pour finalité de faire un tour d'horizon sur la recherche de documents multimédias basée sur leur contenu.

Partie 3

La recherche par le contenu

CHAPITRE CINQUIEME – INDEXATION ET RECHERCHE BASEES SUR LE CONTENU

A l'heure actuelle, de plus en plus de documents multimédias sont stockés dans des bases de données. Afin d'utiliser l'information contenue dans ces documents, il est impératif de disposer d'un système de recherche efficace. Dans les bases de données traditionnelles, l'indexation et la recherche des documents multimédias se basent sur des attributs simples tels qu'une description textuelle du document. C'est ce que l'on appelle l'indexation et la recherche basée sur les mots-clés. Le Data Manager que nous avons abordé au troisième chapitre utilise cette approche pour indexer et rechercher les documents du système. L'inconvénient de ces techniques d'indexation et de recherche est que les attributs textuels ne permettent pas de décrire, par exemple, les caractéristiques d'une image de manière complète et précise. En outre, les systèmes de recherche basés sur les mots-clés ne supportent pas les requêtes basées sur le contenu. Cela signifie qu'il est impossible d'interroger le système à l'aide de requêtes telles que "recherchez, s'il vous plait, toutes les images qui ressemblent à (ou contiennent) celle-ci", en fournissant une image de référence. C'est pourquoi, dans le but de dépasser ces limites des techniques de l'indexation et de la recherche par mots-clés, nous nous tournons vers des techniques d'indexation et de recherche basées sur le contenu.

V.1 – L'extraction automatique de caractéristiques

S'il est facile d'extraire de l'information d'un texte, cela requiert des techniques plus sophistiquées pour les documents audiovisuels, allant de l'analyse d'image à la reconnaissance vocale. Ce sont sur ces techniques que se basent l'indexation et la recherche de documents multimédias basées sur le contenu.

L'indexation et la recherche par le contenu se basent sur les caractéristiques du document multimédia. Il s'agit de l'information que l'on extrait d'un support, pour la représenter d'une façon adéquate, la stocker dans un index et l'utiliser lors du traitement de la requête.
Selon [DJE02], ces caractéristiques sont réparties en deux classes, l'une reprenant les caractéristiques de bas niveau et l'autre reprenant les caractéristiques de haut niveau. Notons qu'aussi bien les caractéristiques de bas niveau que de haut niveau peuvent être représentées à l'aide d'un Schéma de Description MPEG-7 que nous avons vu au deuxième chapitre (section II.3).

V.1.1 – Les caractéristiques de bas niveau

Les caractéristiques de bas niveau (ou primitives) regroupent les mouvements d'objets, la couleur, la texture, la forme, la position dans l'espace des éléments d'une image, les phénomènes spéciaux, le ton pour l'audio, etc.

Dans le cas de l'information visuelle, ces caractéristiques permettent d'effectuer des requêtes comme:

- "Recherchez, s'il vous plait, toutes les séquences vidéo sur lesquelles on peut apercevoir des objets traversant l'image de gauche à droite".
- "Recherchez, s'il vous plait, toutes les photos sur lesquelles on peut voir des petits objets rouges dans le coin supérieur gauche l'image".
- "Recherchez, s'il vous plait, toutes les photos sur lesquelles on peut voir des cercles verts arrangés en rond".
- "Recherchez, s'il vous plait, toutes les photos contenant une zone bleue dans le centre de l'image".
- "Recherchez, s'il vous plait, toutes les photos qui ressemblent à celle-ci".

Les caractéristiques de bas niveau sont objectives et peuvent être dérivés directement de l'image, sans devoir se référer à une base de connaissances extérieure. C'est pourquoi celles-ci peuvent être extraites de manière automatique et calculées efficacement.

Les caractéristiques de bas niveau sont utilisées, par exemple, dans le domaine de la prévention criminelle (identification des empreintes digitales, des visages, etc.), dans le domaine de l'architecture (recherche de plans similaires), dans le domaine de la médecine (recherche de radiographies similaires), etc.

La plupart des systèmes d'indexation supportent les caractéristiques de bas niveau. Mais dans le cas de banques de données multimédia, telles que les bases de données de vidéos et les musées, cela n'est pas suffisant. En effet, dans de tels systèmes, il est impossible d'effectuer des requêtes telles que "recherchez, s'il vous plait, les séquences vidéo dans lesquelles on peut voir Marilyn Monroe dansant au-dessus d'une bouche d'aération".

V.1.2 – Les caractéristiques de haut niveau

Les caractéristiques de haut niveau (ou caractéristiques sémantiques) regroupent plusieurs degrés de sémantique décrits dans le document multimédia. Elles peuvent être divisées en deux groupes: les caractéristiques objectives et les caractéristiques subjectives.

A – Les caractéristiques objectives

Les caractéristiques objectives concernent l'identification d'objets dans une image ou d'actions dans une vidéo. Elles permettent d'effectuer des requêtes similaires aux suivantes:

- "Recherchez, s'il vous plait, toutes les séquences vidéo où l'on peut voir un aigle plongeant sur sa proie".
- "Recherchez, s'il vous plait, toutes les séquences vidéo où l'on peut apercevoir une Harley Davidson noire et rouge".
- "Recherchez, s'il vous plait, toutes les photos sur lesquelles on peut voir une Harley Davidson".
- "Recherchez, s'il vous plait, toutes les photos de l'Atomnium".

Pour pouvoir répondre à ce genre de requêtes, le processus de recherche requiert une connaissance préalable. En effet, pour reconnaître un objet comme étant l'Atomnium, il faut

que le système ait une idée de la structure de l'Atomnium. C'est pourquoi les caractéristiques objectives de haut niveau son annotées de façon manuelle par des indexeurs.
Il est également possible de créer manuellement des classes sémantiques telles que "motos", "monuments", "animaux", "personnes", etc. De cette façon, le système pourrait répondre à des requêtes telles que "recherchez, s'il vous plait, toutes les photos qui ressemblent à celle-ci et qui appartiennent à la classe monuments".

B – Les caractéristiques subjectives

Les caractéristiques subjectives de haut niveau décrivent la signification et le but des objets ou des scènes. Ces caractéristiques peuvent être divisés en classes telles que "événements" (fête nationale), "types d'activités" (pièce de théâtre), "émotions" (sourire), etc. Les systèmes prenant en compte ce type de caractéristiques peuvent répondre à des requêtes telles que "recherchez, s'il vous plait", toutes les photos sur lesquelles on peut voir un enfant sourire", en se basant sur la classe "émotions". Toutefois, cela demande un effort de la part de l'indexeur et du chercheur et peut demander la coopération d'un expert pour interpréter le document et lui associer des caractéristiques.

Dans le cas particulier d'une vidéo, en utilisant des relations temporelles entre les différentes actions et des caractéristiques de haut niveau (objectives et/ou subjectives), on peut effectuer des requêtes telles que "recherchez, s'il vous plait, toutes les séquences vidéo où l'action 1 suit l'action 2".

L'indexation utilisant des caractéristiques de haut niveau est très puissante puisque nous pouvons l'utiliser pour décrire tous les aspects (ou presque) du contenu d'un média, à des niveaux de complexité différents. Toutefois, un processus d'indexation utilisant cette catégorie de caractéristiques demande beaucoup de temps (par exemple, quelques minutes pour indexer une image).
De plus, les caractéristiques de haut niveau que l'on peut assigner à un même document multimédia peuvent être de toutes sortes. Donc, décrire en détail tous les documents multimédia d'un système pourrait être un travail fastidieux qui nécessiterait une assistance automatique. C'est pourquoi il semble intéressant de combiner les deux classes de caractéristiques dans un système d'indexation et de recherche de documents multimédia.

V.1.3 – Les relations entre les caractéristiques de bas niveau et de haut niveau

Supposons que nous disposions d'une base de données contenant des vidéos que nous voulons réutiliser. Nous pourrions indexer les caractéristiques de bas niveau en divisant les séquences vidéos en plans et en générant pour chacun d'entre eux des images-clés représentatives. De cette façon, il est possible de générer automatiquement des story-boards pour chaque vidéo. Nous pourrions ensuite indexer les caractéristiques de haut niveau en annotant les images-clés.

"La clé de cette relation est de construire la sémantique de haut niveau, de manière automatique, sur base de caractéristiques de haut niveau" (cf. [DJE02]).
Une façon de relier les caractéristiques de haut niveau et les caractéristiques de bas niveau est la reconnaissance de formes. Cette dernière consiste à reconnaître et classifier toute une série d'objets extraits d'un support (une image, par exemple). Elle se base sur les caractéristiques de l'objet ciblé telles que la couleur, la forme ou la texture, ainsi que sur des méta-informations telles que la position dans l'espace, les relations spatiales avec les autres objets et l'arrière plan de l'image.

La reconnaissance de forme se base sur les trois principes suivants:

- Identifier des classes d'objets.
- Indiquer des zones d'image pouvant comprendre des exemples de ces objets.
- Donner un exemple de mécanismes pour valider la présence d'un objet

Une autre approche permettant de relier les caractéristiques des deux niveaux est la découverte de connaissances. Cela consiste, par exemple, à sélectionner certaines zones d'une image, de les annoter sémantiquement et d'appliquer aux zones répondant aux mêmes caractéristiques des annotations similaires.
Un autre exemple de cette approche est la découverte de concepts. Imaginons qu'un système de recherche d'images traite une requête qu'il a déjà rencontrée plusieurs fois. Il pourrait s'en souvenir et créer un concept composé des caractéristiques de la requête et des images associées. Ensuite le système proposerait à l'utilisateur de valider le concept en lui assignant une étiquette sémantique (telle que "t-shirt rouge"). Si le concept est validé, il est stocké, en même temps que l'image, dans la base de données. Ensuite, un thesaurus visuel des concepts est créé dans lequel chaque concept est relié à une description étiquetée par l'utilisateur, aux caractéristiques qui lui sont associées et aux images. De cette façon, les concepts pourront être utilisés dans des requêtes futures.

V.2 – MPEG-7 pour l'indexation et la recherche basées sur le contenu

Nous l'avons vu, MPEG-7 fournit toute une série d'outils permettant de décrire le contenu de documents audiovisuels.
On pourrait imaginer un système d'indexation et de recherche de documents multimédia basé sur leurs descriptions MPEG-7. Les deux projets décrits dans cette section mettent en œuvre un tel système.

V.2.1 – Le projet MADIS

Le projet MADIS ([MADIS]), réalisé en collaboration avec le Centre de Recherche Informatique de Montréal (CRIM) et l'Office National du Film (ONF) du Canada, vise un système d'indexation et de recherche de documents multimédia basé sur les descriptions MPEG-7.

Ce système est accessible sur l'Internet et contient les éléments suivants:

- Une base de données multimédia (pour les documents multimédias).
- Une base de données MPEG-7 (pour les descriptions MPEG-7).
- Un système d'extraction de caractéristiques.
- Un serveur d'application Web pour le traitement des requêtes.

Comme nous pouvons le voir à la figure V.1, le système d'extraction des caractéristiques traite les documents multimédia et en extrait les différentes caractéristiques afin de produire les fichiers de description MPEG-7. Ce traitement se fait offline et est adapté au type du support dont il faut extraire l'information. Cela signifie que les algorithmes utilisés pour extraire les caractéristiques d'une image ne sont pas les mêmes que ceux utilisés dans le cas d'une vidéo ou d'un son.

Une fois les descriptions créées, elles sont stockées dans la base de données MPEG-7, alors que les documents multimédia sont stockés dans la base de données multimédia.

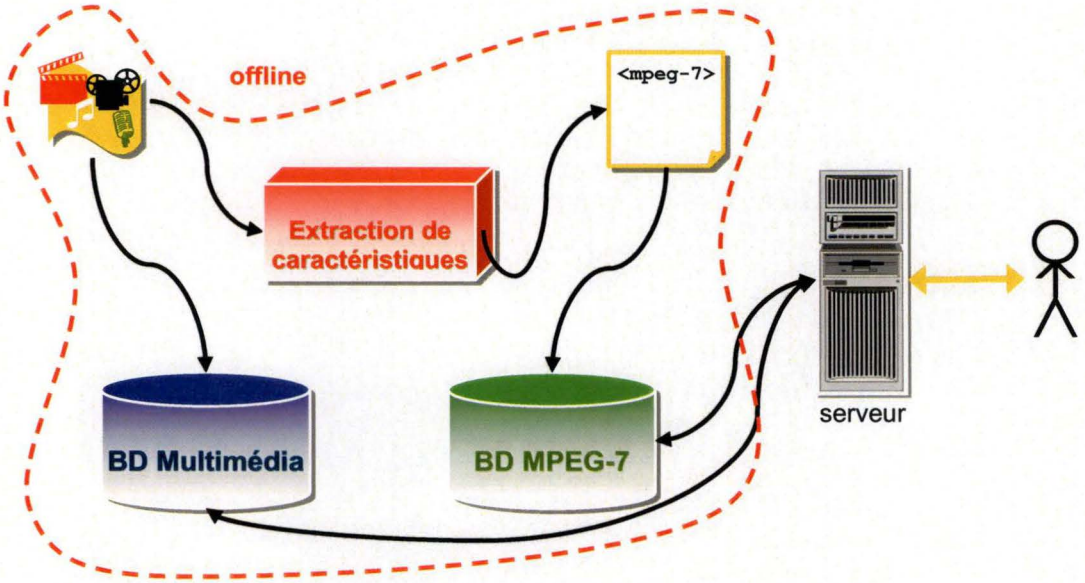


Figure V.1. Architecture du système MADIS.

Le serveur d'application Web a pour finalité de saisir les requêtes entrées par les utilisateurs, de retourner les résultats pertinents pour ces requêtes et d'acheminer la transmission du document multimédia pour les utilisateurs. Il est composé d'un moteur de recherche et d'un outil permettant de classer les résultats.

L'utilisateur du système peut interroger celui-ci à l'aide de requêtes par mots-clés ou encore par document type.
Dans le premier cas, l'utilisateur précise un ou plusieurs éléments qu'il désire trouver dans le document (comme par exemple "chaussettes rouges") et le système cherche après tous les documents ayant un rapport avec le ou les mots-clés introduits.
Dans le deuxième cas, l'utilisateur sélectionne un document du système qui ressemble à celui qu'il désire retrouver et le système recherche tous les documents présentant les mêmes caractéristiques. On pourrait également imaginer que l'utilisateur fournisse lui-même une image, par exemple, similaire à celle qu'il désire retrouver.

Le Data Manager, visé au troisième chapitre, est également proche de ce genre de systèmes: en ajoutant à celui-ci un outil permettant d'extraire les caractéristiques des documents multimédia, il serait possible de réaliser les mêmes fonctions de recherche. En effet, l'état actuel du Data Manager permettant d'indexer des documents XML, il convient également à l'indexation d'éléments contenus dans une description MPEG-7. Dès lors, l'outil d'extraction obtenu permettrait de créer ces descriptions MPEG-7 qui seraient traitées de la même façon que les descripteurs XML. L'utilisateur aurait alors la possibilité d'interroger le système à l'aide de requêtes constituées de mots-clés. En outre, l'outil d'extraction pourrait être utilisé lors de requêtes par document-type: l'information contenue dans ce document serait extraite afin d'être comparée à l'information des documents du système. A cette fin, les différents éléments de la description du document type constitueraient une requête qui serait soumise au moteur de recherche du Data Manager.

V.2.1 – Le projet Net Imager

MPEG-7 Net Imager est un projet réalisé par des étudiants de l'Ecole Centrale de Lyon (option informatique), en 2001[1]. Il s'agit d'un encodeur et décodeur de documents multimédia au format MPEG-7 qui permet de rechercher des documents multimédia et de les indexer de manière automatique. Le but du projet était de réaliser un encodeur automatique MPEG-7, pour les contenus multimédia, ainsi qu'un décodeur MPEG-7 permettant une recherche à partir d'un navigateur. Comme l'indique le nom du projet, les documents multimédia pris en compte dans Net Imager sont les images.

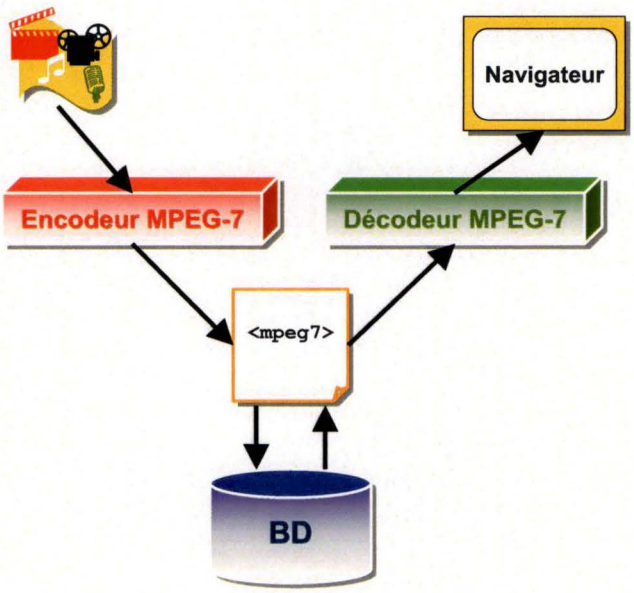


Figure V.2. Architecture de Net Imager.

La figure V.2 illustre la manière dont les étudiants ont schématisé leur travail. Les documents multimédias sont stockés dans une base de données et décrits à l'aide de l'encodeur MPEG-7. Les descriptions sont ensuite stockées dans une autre base de données pour qu'elles puissent être retrouvées par la suite. Une interface Web donne accès à un moteur de recherche qui permettant d'extraire les informations souhaitées. Le moteur de recherche interroge la base de données des descriptions à l'aide de requêtes XML et l'interface Web permet au navigateur d'interpréter les fiches les descriptions (qui sont des fichiers XML).
Les descriptions MPEG-7 de Net Imager sont stockées dans une base de données relationnelle. C'est pourquoi Net Imager possède un transcripteur permettant de transformer le code XML en tuples afin de l'insérer dans la base de données. Nous pourrions imaginer d'utiliser directement une base de données XML, ce qui éliminerait cette opération de transcription supplémentaire.
Le moteur de recherche XML du projet consiste en un transcripteur de requêtes XML en requêtes SQL.

[1] L'adresse de la page du projet est http://perso.wanadoo.fr/bstrazza/projets/mpeg7/ (accédé le 14/08/2004).

Encodage MPEG-7

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<MPEG7>
   <EncodedBy>Projet MPEG-7 Net Imager</EncodedBy>
   <Annotation xml:lang='fr'>
      <Who>Lucas</Who>
      <WhatObject>vélo</WhatObject>
      <WhatAction>roule</WhatAction>
      <Where>étang</Where>
      <When>juillet 2003</When>
      <Why>anniversaire</Why>
      <TextAnnotation>
        Lucas essaie son nouveau vélo qu'il a reçu pour son anniversaire
      </TextAnnotation>
   </Annotation>
   <Creator>
      <Individual>
         <GivenName>Gérard</GivenName>
         <FamilyName>Mensoif</FamilyName>
      </Individual>
   </Creator>
   <MediaCoding>
      <FrameWidth>71</FrameWidth>
      <FrameHeight>106</FrameHeight>
      <CompressionFormat></CompressionFormat>
   </MediaCoding>
   <MediaInstance>
      <InstanceLocator>
         <MediaURL>D:\picture\bicycle.jpg</MediaURL>
      </InstanceLocator>
   </MediaInstance>
   <ColorQuantizationType colorSpace='RGB'>
      <Bins>
        -14671840;-14671776;-14671712;-14671648;-14655456;-14655392;-14655328;-14655264;
        -14639072;-14639008;-14638944;-14638880;-14622688;-14622624;-14622560;-14622496;
        -10477536;-10477472;-10477408;-10477344;-10461152;-10461088;-10461024;-10460960;
        -10444768;-10444704;-10444640;-10444576;-10428384;-10428320;-10428256;-10428192;
        -6283232;-6283168;-6283104;-6283040;-6266848;-6266784;-6266720;-6266656;
        -6250464;-6250400;-6250336;-6250272;-6234080;-6234016;-6233952;-6233888;
        -2088928;-2088864;-2088800;-2088736;-2072544;-2072480;-2072416;-2072352;
        -2056160;-2056096;-2056032;-2055968;-2039776;-2039712;-2039648;-2039584
      </Bins>
   </ColorQuantizationType>
   <ScalableColorType numberOfCoefficients='63'>
      <Coefficients>000000000110000101011101100110100011001001000000000100111
        10111110</Coefficients>
   </ScalableColorType>
   <ColorStructureType colorQuant='64'>
      <Histogram>
        0.8113952;0.33964646;0.0;0.0;0.18181819;0.42771465;0.021464646;0.0;0.0;0.0;0.0;
        0.0;0.0;0.0;0.0;0.0;0.20817551;0.030303031;0.0;0.0;0.3989899;0.6987058;
        0.37626263;0.0;0.080650255;0.20880681;0.32544193;0.0;0.0;0.0;0.0;0.0;0.01010101;
        0.0;0.0;0.0;0.11063763;0.23847854;0.026515152;0.0;0.10969066;0.37594697;
        0.43118685;0.02935606;0.038036615;0.069286615;0.052714646;0.01010101;0.0;0.0;
        0.0;0.0;0.0;0.05224116;0.0;0.0;0.09217171;0.14567551;0.11237374;0.0;0.08017677;
        0.1144255;0.07544192;0.08270202
      </Histogram>
   </ColorStructureType>
</MPEG7>
```

Figure V.3. Instance du Schéma de Description produit par Net Imager.

A – Le Schéma de Description utilisé dans Net Imager

Le Schéma de Description utilisé dans Net Imager pour décrire les images du système comporte quatre parties: la description textuelle, la description des personnes, la description du point de vue média et la description visuelle. Il s'agit des éléments qui sont extraits de manière automatique par l'encodeur MPEG-7 de Net Imager. A titre d'exemple, la figure V.3 illustre une instance du Schéma de Description. La fiche XML de l'illustration a été générée par Net Imager.

La description textuelle

Cette partie concerne:

- L'attribut **`langage`** tel qu'il est défini dans XML, qui permettra de spécifier la langue utilisée pour les annotations: `xml:lang`
- Le Descripteur **`ControlledTerm`** qui sera utilisé dans les Schémas de Description `Annotation` et `MediaCoding`:

```
<DType name="controlledTerm" datatype="string">
    <attribute name="CSName" datatype="string" required="false"/>
    <attribute name="CSTermId" datatype="string" required="false"/>
    <attribute name="CSLocation" datatype="uri" required="false"/>
</DType>
```

- Le Schéma de Description **`Annotation`** permettant de décrire les six W's d'un média:

```
<DType name="TextAnnotation" datatype="string"/>

<DSType name="Annotation">
    <attribute name="id" datatype="ID"/>
    <DTypeRef name="Who" type="controlledTerm" minOccurs="0"/>
    <DTypeRef name="WhatObject" type="controlledTerm" minOccurs="0"/>
    <DTypeRef name="WhatAction" type="controlledTerm" minOccurs="0"/>
    <DTypeRef name="Where" type="controlledTerm" minOccurs="0"/>
    <DTypeRef name="When" type="controlledTerm" minOccurs="0"/>
    <DTypeRef name="Why" type="controlledTerm" minOccurs="0"/>
    <DTypeRef type="TextAnnotation" minOccurs="0"/>
</DSType>
```

La description des personnes

Dans cette partie du Schéma de Description, on trouve:

- Le Schéma de Description **`Individual`** qui permet de décrire un individu à l'aide de son nom et son prénom et qui sera utilisé dans le Schéma de Description `Person`:

```
<DType name="FamilyName" datatype="string"/>
<DType name="GivenName" datatype="string"/>

<DSType name="Individual">
    <attribute name="id" datatype="ID"/>
    <DTypeRef type="FamilyName"/>
    <DTypeRef tyê="GivenName"/>
</DSType>
```

- Le Schéma de Description **Person** qui permettra de décrire le créateur ou l'éditeur de l'image:

```
<DSType name="Person">
   <attribute name="id" datatype="ID"/>
   <choice>
      <DSTypeRef type="Individual"/>
   </choice>
</DSType>

<DstypeRef name="Creator" type="Person"/>
<DstypeRef name="Publisher" type="Person"/>
```

La description du point de vue média

Cette partie du Schéma de Description concerne:

- Le Schéma de Description **MediaCoding** permettant de préciser la taille de l'image et son format de compression:

```
<DType name="FrameWidth" datatype="non-negative-integer"/>
<DType name="FrameHeight" datatype="non-negative-integer"/>

<DSType name="MediaCoding">
   <attribute name="id" datatype="ID">
   <DTypeRef type="FrameWidth" minOccurs="0"/>
   <DTypeRef type="FrameHeight" minOccurs="0"/>
   <DTypeRef name="CompressionFormat" type="controlledTerm"
     minOccurs="0">
</DSType>
```

- Le Schéma de Description **MediaInstance** qui permet de préciser le nom ou l'adresse de l'image:

```
<DType name="MediaName" datatype="string"/>
<DType name="InstanceLocator">
   <choice>
      <DTypeRef type="MediaURL"/>
      <DTypeRef type="MediaName"/>
   </choice>
</DType>

<DSType name="MediaInstance">
   <attribute name="id" datatype="ID"/>
   <DTypeRef type="InstanceLocator"/>
</DSType>
```

La description visuelle

Dans cette partie du Schéma de Description, on trouve:

- Le Schéma de Description **ColorQuantization** qui permet de décrire la quantification des couleurs de l'image:

```
<DSType name="ColorQuantizationType">
   <attribute name="colorSpace" type="string"/>
   <element name="Bins" type="string"/>
</DSType>
```

- Le Schéma de Description **`ScalableColor`**:

```
<DSType name="ScalableColorType">
   <attribute name="numberOfCoefficients" type="unsigned3"/>
   <sequence>
      <element name="Coefficients" type="string"/>
   </sequence>
</DSType>
```

- Le Schéma de Decription **`ColorStructure`** permettant de décrire la structure des couleurs de l'image:

```
<DSType name="ColorStructureType">
   <attribute name="colorQuant" type="unsigned3" use="required"/>
   <element name="Histogram" type="string"/>
</DSType>
```

B – L'encodeur MPEG-7 de Net Imager

L'encodeur de Net Imager se compose d'un encodeur visuel et d'un encodeur conceptuel. Le premier permet l'extraction automatique de différents éléments visuels alors que le second permet d'annoter les images et d'en préciser le créateur. Donc, l'encodeur visuel concerne des caractéristiques de bas niveau et l'encodeur conceptuel concerne des caractéristiques de haut niveau.

L'encodeur visuel

L'encodeur visuel de Net Imager est une interface permettant à un utilisateur d'extraire des caractéristiques visuelles d'une série d'images de façon automatique. La figure V.4 nous donne un aperçu de cette interface.

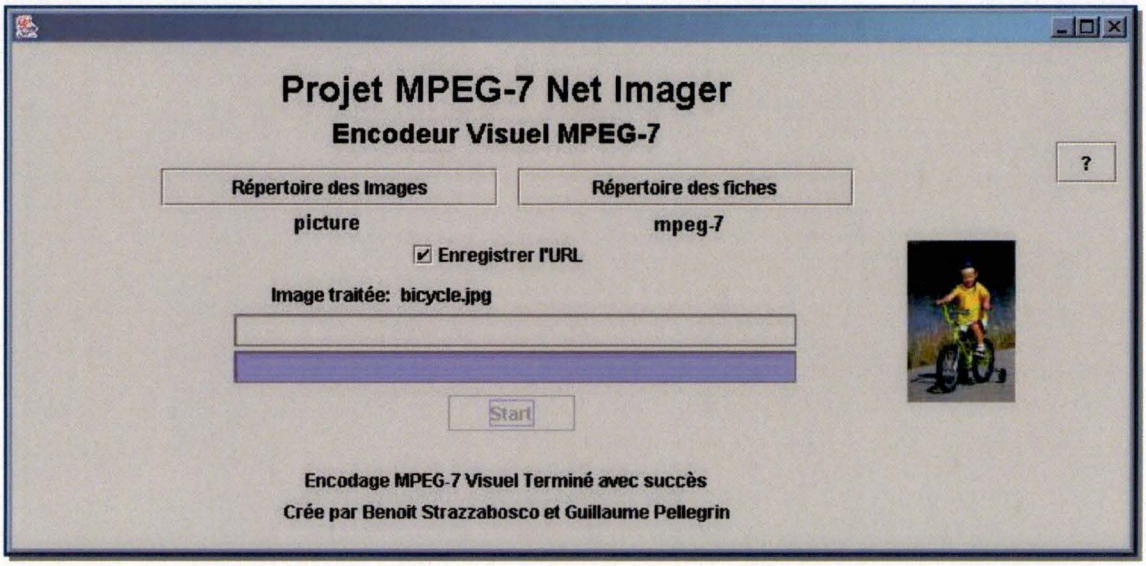


Figure V.4. L'interface de l'encodeur visuel MPEG-7 de Net Imager.

L'encodeur visuel de Net Imager offre à l'utilisateur un moyen de préciser un répertoire des images et un répertoire des fiches. Le premier correspond à l'endroit où

l'utilisateur a placé les images qu'il désire encoder et le second correspond à l'endroit où il désire retrouver les descriptions générées. En outre, l'encodeur donne le choix à l'utilisateur d'enregistrer l'adresse de l'image sur le disque ou le nom de celle-ci. Une fois que l'utilisateur a précisé ces paramètres, il peut enclencher l'encodage des images.
Cet outil se charge de donner des valeurs aux éléments `MediaCoding`, `MediaInstance`, `ColorQuantizationType`, `ScalableColorType` et `ColorStructureType`.

L'encodeur conceptuel

L'encodeur conceptuel de Net Imager est une interface permettant à un utilisateur d'annoter une image à la main et de préciser les nom et prénom de son créateur. Nous pouvons voir un aperçu de cette interface à la figure V.5.

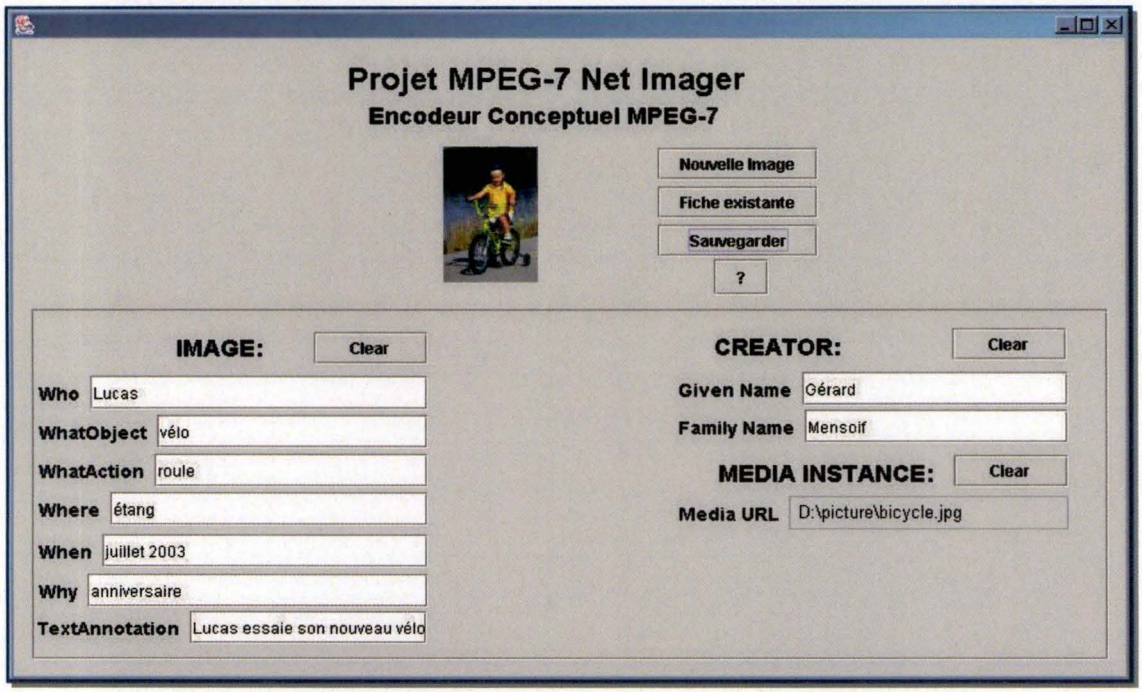


Figure V.5. L'interface de l'encodeur conceptuel MPEG-7 de Net Imager.

Cet outil permet donc de remplir les valeurs des éléments `Creator` et `Annotation`. A cette fin, l'interface propose à l'utilisateur de remplir sept champs d'annotation:

- `Who` permet à l'utilisateur de citer la ou les personnes qu'il voit sur l'image (par exemple *Lucas*).
- `WhatObject` permet à l'utilisateur de citer les objets qui se trouvent sur l'image (par exemple un *vélo*).
- `WhatAction` permet à l'utilisateur de décrire ce qui se passe sur l'image en termes d'action (par exemple *roule*).
- `Where` permet à l'utilisateur de préciser le lieu de la photo (par exemple *l'étang* du village).
- `When` permet à l'utilisateur de préciser le moment de la photo (par exemple en *juillet 2003*).

- `Why` permet à l'utilisateur de préciser pourquoi la photo à été prise (par exemple à l'occassion d'un *anniversaire*).
- `TextAnnotation` permet à l'utilisateur de fournir des informations supplémentaires sur l'image (par exemple "*Lucas essaie son nouveau vélo qu'il a reçu pour son anniversaire*").

En outre, l'interface donne la possibilité à l'utilisateur de citer les nom et prénom du créateur de la photo à l'aide des champs `Given Name` et `Family Name`.
Les caractéristiques générées par cet encodeur peuvent soit s'ajouter à une description existante (i.e. contenant déjà des caractéristiques visuelles), soit former une nouvelle description (ne contenant que des caractéristiques conceptuelles). Le choix de la finalité de l'encodage est donné à l'utilisateur à l'aide des boutons **"Fiche Existante"** et **"Nouvelle Image"**, respectivement.

Nous l'avons compris, le principe général mis en œuvre dans les projets que nous venons d'étudier est applicable à n'importe quel document multimédia. En effet, il suffit d'adapter l'outil (et donc, les algorithmes) d'extraction des caractéristiques du document en fonction de celui-ci. Ainsi, si le système a pour but de gérer des documents audio, il comprendra un outil permettant d'extraire des caractéristiques sonores (par exemple, le ton, ou la mélodie). En revanche, si l'objectif est de gérer des documents visuels, il comprendra un outil permettant d'extraire des caractéristiques visuelles (par exemple, les couleurs ou les formes). Dans le chapitre suivant, nous allons nous pencher sur le cas particulier de la recherche d'images fixes basée sur le contenu.

CHAPITRE SIXIEME – LE CAS PARTICULIER DES IMAGES

VI.1 – La recherche d'images basée sur le contenu

Comme nous pouvons aisément l'imaginer, la recherche d'image basée sur le contenu (RIBC ou CBIR pour *Content-Based Image Retrieval*) est une technique permettant de retrouver des images sur base de caractéristiques automatiquement dérivables de celles-ci[2].
En général les systèmes de recherche d'images basée sur le contenu travaillent sur les caractéristiques de bas niveau: ils calculent des informations sur la couleur, la texture ou la forme de l'image. Ces informations sont soit stockées (dans le cas de l'indexation), soit utilisées pour effectuer une requête (dans le cas de la recherche). L'extraction des caractéristiques de haut niveau reste un domaine difficile qui est l'objet de nombreuses recherches (e.g. [CAM99] et [EAGR99]).

VI.1.1 – Fonctionnement du système de recherche d'images basée sur le contenu

Selon [VETA00], la plupart des systèmes RIBC peuvent être décrits conceptuellement, comme illustré à la figure VI.1. L'interface utilisateur se compose généralement d'un outil de formulation de requêtes et d'un outil de présentation des résultats. Lorsqu'il effectue une requête, l'utilisateur a une idée du type d'images qu'il recherche. Il peut par exemple vouloir retrouver dans ces images des éléments tels que:

- Une combinaison particulière de couleur, texture et forme (par exemple, des ronds bleus).
- Une disposition particulière d'objets donnés (par exemple, un oiseau sur une branche).
- Un type d'événement particulier (par exemple, un match de basket-ball).
- Un événement, lieu ou individu particuliers (par exemple, le premier anniversaire de la princesse Elizabeth).
- Une émotion particulière (par exemple, la gaieté).
- Des métadonnées telles que l'auteur, la date de création, le lieu de création de l'image, etc. (par exemple, de Gérard Mensoif ou créée en 2003).

Quelque soit le type d'images recherché, l'utilisateur peut interroger la base de données de différentes façons:

- Parcourir les images de la base de données une par une.
- Spécifier l'image en termes de mots-clés.
- Spécifier l'image en termes de caractéristiques extraites des images de la base de données.
- Fournir une image similaire à celle recherchée.
- Faire un croquis de l'image recherchée.

[2] Une lecture intéressante pour la recherche d'images par le contenu est la thèse de E. Loupias ([LOU00]).

Dans les deux derniers cas, des caractéristiques de l'image ou croquis-type sont extraites, afin d'être comparées avec les caractéristiques des images de la base de données. Dans ce cas, il est impératif que les images-type subissent le même traitement que les images de la base de données, pour que les caractéristiques à comparer aient une présentation similaire.

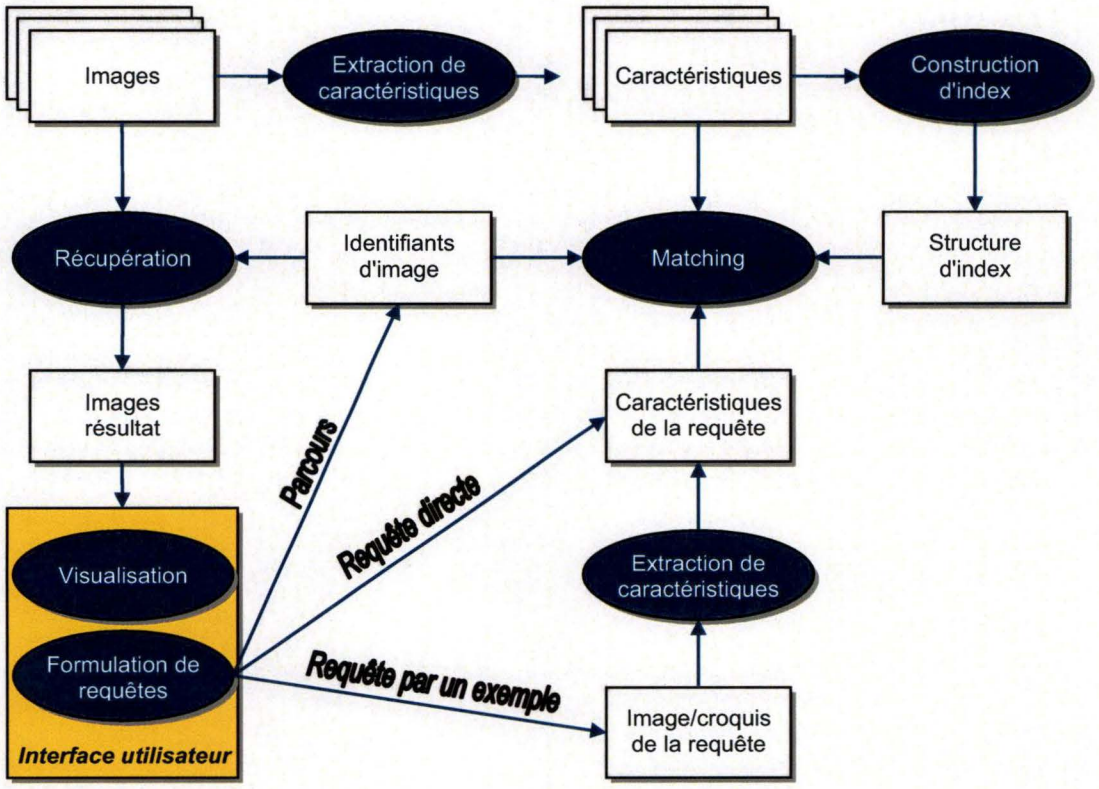


Figure VI.1. Architecture d'un Système de Recherche d'Image basée sur le Contenu.

En général, les caractéristiques utilisées lors des requêtes concernent la couleur, la texture, la forme et l'organisation spatiale de l'image ainsi que les visages qu'elle contient.

A – La recherche par la couleur

Les caractéristiques concernant les couleurs contenues dans l'image peuvent être directement dérivées de l'intensité des pixels, à l'aide d'un histogramme de couleurs.

Histogramme de couleurs

Un histogramme de couleurs est un graphique permettant de représenter la proportion de pixels pour chaque couleur de l'image. Comme nous pouvons le voir à la figure VI.2, l'abscisse de l'histogramme reprend les niveaux d'intensité de la couleur du plus clair au plus foncé et l'ordonnée nous donne le nombre de pixels de l'image. De cette façon, pour représenter l'histogramme de couleurs d'une image en 256 niveaux de gris, l'abscisse contiendra 256 valeurs différentes. Pour une image dans l'espace de couleur RGB, l'histogramme de couleurs sera composé de trois histogrammes reprenant respectivement les valeurs des couleurs rouge, verte et bleue de l'image.

La construction d'un histogramme de couleurs se fait simplement en parcourant les pixels de l'image de gauche à droite et de haut en bas et en inscrivant leur niveau d'intensité.

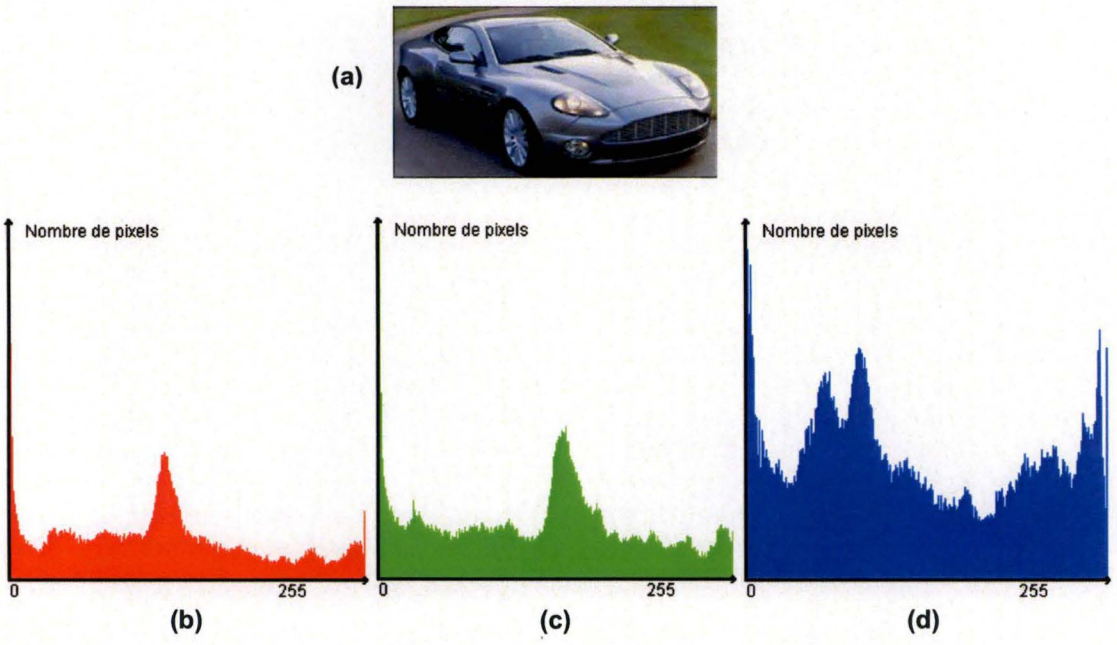


Figure VI.2. Histogramme de couleurs d'une image. **(a)** Image originale. **(b)** Histogramme de la composante rouge. **(c)** Histogramme de la composante verte. **(d)** Histogramme de la composante bleue.

L'idée de la recherche basée sur la couleur est de calculer, lors de l'indexation et pour chaque image du système, un histogramme de couleurs et de stocker celui-ci dans la base de données. Lors de sa recherche, l'utilisateur peut alors spécifier la proportion des couleurs qu'il désire retrouver dans l'image en effectuant une requête telle que "retrouvez, s'il vous plait, toutes les images contenant 75% de bleu roi et 15% de rouge". Le système recherche les histogrammes répondant le plus à ces proportions et renvoie à l'utilisateur les images correspondantes. L'utilisateur pourrait également fournir au système une image similaire à celles qu'il désire retrouver. Dans ce cas, l'histogramme de couleurs de cette image est calculé et comparé aux différents histogrammes de couleurs de la base de données. Le système fournit alors à l'utilisateur les images ayant un histogramme similaire à celui de l'image-type.

B – La recherche par la texture

Malgré les nombreuses tentatives de définition, le concept de *texture* reste relativement flou. Le dictionnaire[3] définit le terme texture comme suit:

> *« Arrangement, disposition (des éléments d'une matière), agencement des parties des éléments d'une œuvre, d'un tout. »*

Comme nous le fait remarquer [BJL03], le concept de texture est donc lié à la notion de répartition spatiale des éléments constitutifs d'une image.

[3] Le nouveau Petit Robert, édition de 1994.

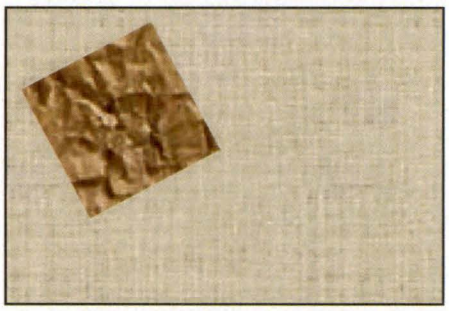


Figure VI.3. Exemple d'une image composée de deux textures.

La recherche d'images basée sur la similarité de la texture est utile pour distinguer des zones d'image ayant des couleurs similaires (par exemple le ciel et la mer, ou encore les deux formes de la figure VI.3). Il y a plusieurs techniques permettant d'extraire les caractéristiques de la texture. L'approche statistique utilise l'intensité de chaque pixel d'une image et applique sur les pixels diverses formules statistiques pour calculer ces caractéristiques.
Selon [SGU04], certains systèmes (comme QBIC, que nous aborderons plus loin) utilisent trois mesures de la texture: la rugosité (*coarseness*), le contraste et l'orientation. D'autres méthodes d'analyse de texture utilisent les filtres de Gabor et les fractales.

L'idée de la recherche par la texture est de calculer, lors de l'indexation, différents paramètres concernant la texture des images et de les stocker. Par la suite, le système permettra à l'utilisateur de sélectionner une texture parmi une palette de textures ou de fournir une image-type. Les caractéristiques de la texture sont alors extraites et comparées aux paramètres présents dans le système. Les images correspondant aux mesures les plus proches de la texture en entrée seront renvoyées à l'utilisateur.
On pourrait également imaginer un système où l'utilisateur décrit lui même la texture à laquelle il pense. Dans ce cas il faudrait associer chaque texture présente dans le système à un terme qui la décrirait. Mais la texture n'étant pas un concept bien défini, un tel système ne donnerait pas de bons résultats.

C – La recherche par la forme

De toutes les caractéristiques de bas niveau, c'est probablement la forme de l'image (ou les formes contenues dans celle-ci) qui a le plus de pertinence pour la recherche par le contenu. De plus, contrairement à la texture, la forme est un concept clairement défini. Par exemple, lorsque l'on pense à un carré, on imagine une forme composée de quatre côtés, c'est une notion universelle. En revanche, tout le monde ne s'imaginera pas une texture granuleuse de la même façon.
En 1987, Biederman a développé une théorie de reconnaissance par composants (*Recognition-by-component*) selon laquelle *"la reconnaissance des objets repose sur la perception d'éléments géométriques de base à partir desquels on peut construire un objet"*.
Le système visuel humain identifie un objet par ses contours, qu'il s'agisse de contours extérieurs ou intérieurs, en se basant sur la différence d'intensité des pixels. Comme nous l'avons vu au deuxième chapitre (section II.3.2), les outils permettant de décrire une forme se basent soit sur son contour, soit sur les zones qui la composent.

L'idée est donc ici de calculer, pour chaque objet identifié dans chaque image du système, un certain nombre de caractéristiques de sa forme (indépendamment de la taille ou de la direction). Ces caractéristiques sont stockées dans le système. De cette façon, l'utilisateur pourra interroger le système à l'aide de requêtes dans lesquelles il cite les formes

qu'il désire retrouver dans l'image qu'il recherche. Une autre solution est de fournir au moteur de recherche une (image contenant la) forme. Dans ce cas, le système calcule les caractéristiques de la forme, recherche les images offrant les mêmes caractéristiques et les renvoie à l'utilisateur.

D – La recherche par l'organisation spatiale

La recherche par l'organisation spatiale consiste à retrouver des objets en fonction de leur position relative dans une image. L'idée est donc de calculer la disposition des objets contenus dans l'image lors de l'indexation. Par après, l'utilisateur pourra interroger le système en effectuant une requête telle que "retrouvez, s'il vous plait, toutes les images sur lesquelles on peut voir des étoiles arrangées dans un cercle". Nous pourrions également imaginer que l'utilisateur dessine la position relative des objets qu'il désire retrouver dans l'image. Le système fournirait alors à l'utilisateur les images correspondant le plus à l'organisation spatiale désirée.

E – La recherche par la reconnaissance de visages

La recherche par la reconnaissance de visages consiste à retrouver les images contenant un visage particulier. L'idée est d'extraire les visages contenus dans les images du système, d'en calculer les caractéristiques (le contour, la forme de la bouche, la distance entre les deux yeux, etc.) et de stocker celles-ci. Dès lors, l'utilisateur a la possibilité de fournir au système la photographie d'un visage en lui demandant de retrouver toutes les images contenant ce visage. Ensuite, le système calculerait les caractéristiques du visage afin de rechercher les images contenant les visages ayant des caractéristiques similaires. Nous pourrions également imaginer un système où l'utilisateur fournit le nom de la personne dont il désire retrouver la photographie. Dans ce cas, il faut que le système contienne une table de correspondance entre les visages du système et les noms des personnes qu'ils représentent. Evidemment, la création de cette table demanderait une intervention extérieure. Nous pourrions imaginer, par exemple, qu'à chaque fois qu'un utilisateur trouve une image contenant le visage qu'il recherche, le système lui donne la possibilité de fournir le nom de la personne. Cette approche sera approfondie dans la section VI.2.

A l'heure actuelle, il existe différents systèmes de recherche d'images basée sur le contenu. Parmi les systèmes commerciaux, on trouve l'outil QBIC que nous allons décrire dans la section suivante.

VI.1.2 – Un système de recherche d'images basée sur le contenu: QBIC

QBIC[4] est un système permettant de traiter, d'organiser et de parcourir les images dans une base de données de taille importante. Il permet à l'utilisateur d'effectuer des requêtes basées sur le contenu visuel d'une image. Les propriétés sur lesquelles le système se base sont, par exemple, le pourcentage de couleurs, la répartition des couleurs et les textures qui apparaissent dans l'image. Les requêtes que le système permet d'effectuer utilisent les propriétés visuelles de l'image. Il est donc possible de rechercher des images, en fonction de leurs couleurs, leurs textures et leurs positions, sans utiliser de mots. Pour

[4] QBIC™ (ou Query By Image Contents) a été développé par IBM. Vous trouverez la page Web de QBIC dans [QBIC].

obtenir de meilleurs résultats, le système permet de combiner les requêtes avec des mots-clés ou du texte.

QBIC permet à l'utilisateur de réaliser une recherche d'images par mots-clés, en introduisant le nom de l'auteur, le titre du média, etc. Le système présente alors à l'utilisateur des images miniatures correspondant à l'auteur ou au support spécifié. L'utilisateur a dès lors la possibilité d'afficher une image dans sa taille réelle.

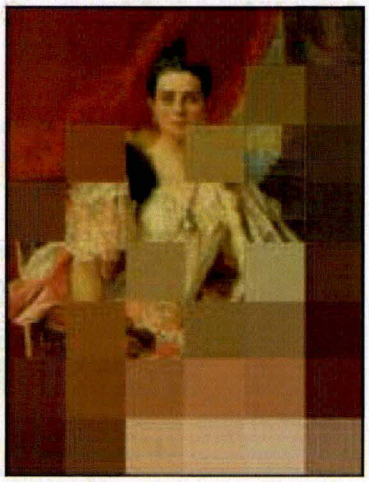


Figure VI.4. QBIC compare la grille de couleurs avec les images de la base de données.

A titre d'exemple, le Musée de l'Hermitage de Saint-Pétersbourg (URSS) utilise le moteur de QBIC pour la recherche d'images d'œuvres d'art sur son site Web[5]. Ce site permet à l'utilisateur de retrouver les œuvres d'art en sélectionnant des couleurs sur une palette, ou en dessinant les formes sur un tableau. Le tableau est interprété comme une grille constituée de zones de couleurs (cf. image VI.4) qui est comparée aux images contenues dans la base de données. QBIC permet également d'affiner la recherche en demandant au système de retrouver toutes les œuvres d'art ayant les mêmes attributs visuels que celles fournies par le système.

A – La recherche par couleurs de QBIC

La recherche par couleurs (*Colour Search*) de QBIC permet au système de repérer les images d'œuvres d'art qui correspondent aux couleurs que l'utilisateur spécifie. A cette fin, QBIC fournit un outil qui permet à l'utilisateur de sélectionner des couleurs à partir d'une palette, d'en définir les proportions et d'exécuter la recherche.
A titre d'exemple, la figure VI.5 illustre une requête par couleur effectuée sur le site Web du Musée de l'Hermitage. Dans un premier temps, l'utilisateur choisit donc des couleurs dans la palette, décrit la proportion voulue de ces couleurs dans l'œuvre d'art recherchée et exécute la recherche. Ensuite, le système renvoie une série de résultats correspondant le plus à la requête initiale. Ces résultats sont classés par ordre de pertinence. Dès lors, l'utilisateur a la possibilité d'afficher l'œuvre d'art qu'il désire, parmi ces résultats, ainsi que la description de celle-ci (titre, auteur, type d'œuvre, etc.). A ce moment, l'utilisateur peut également affiner sa recherche en demandant les œuvres d'art similaires à celle qu'il a sélectionné (par exemple, les œuvres du même artiste, du même style, provenant de la même région, etc.). Les résultats de cette recherche par similarité sont présentés de la même manière que les résultats par couleur.

[5] Vous trouverez le site web du Musée de l'Hermitage à l'adresse http://www.hermitagemuseum.org.

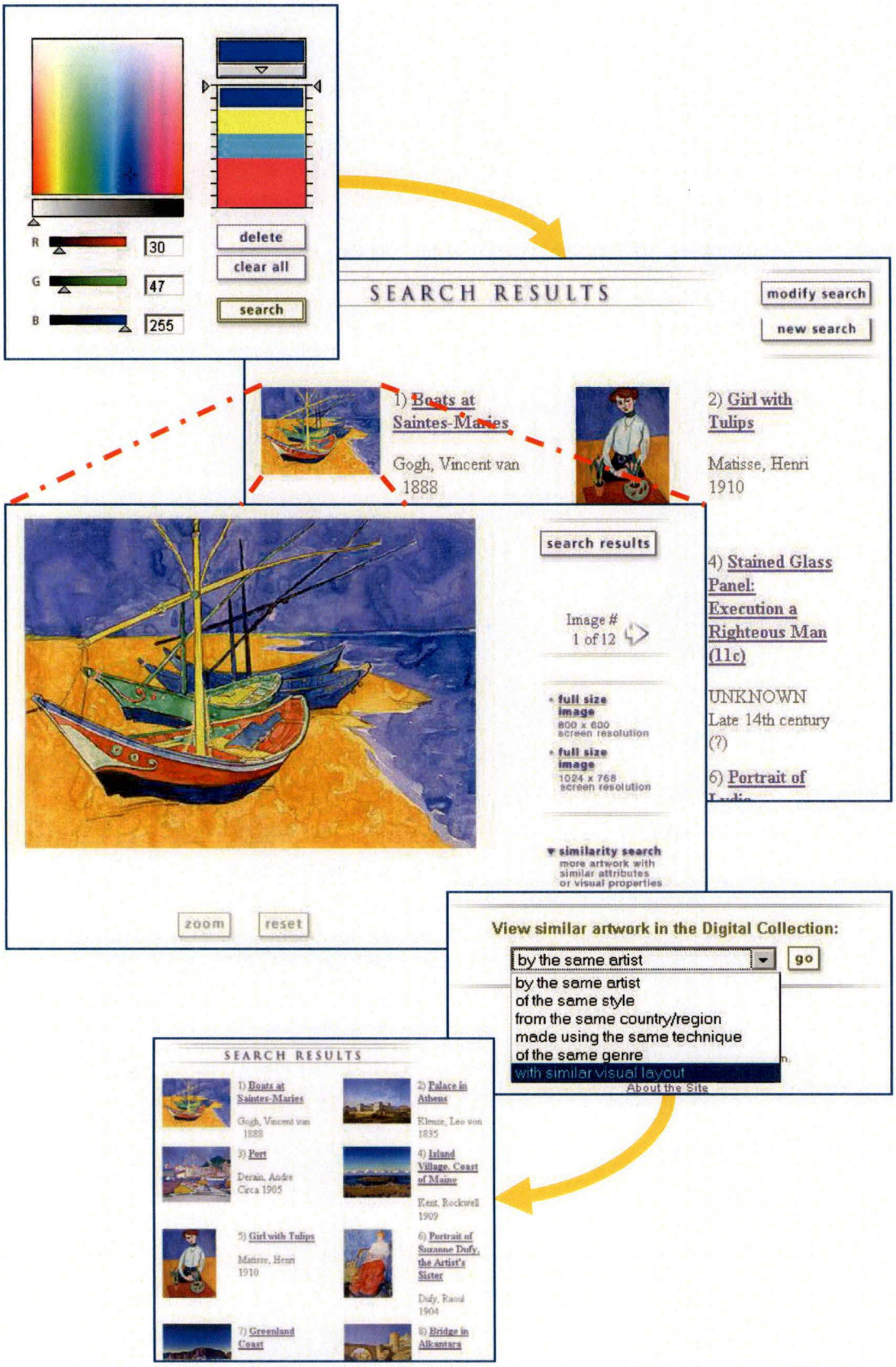


Figure VI.5. La recherche par couleurs de QBIC.

B – La recherche par plan de QBIC

La recherche par plan (*Layout Search*) de QBIC permet à l'utilisateur de dessiner l'œuvre d'art qu'il recherche. Il a donc à sa disposition un outil qui lui permet de disposer, sur un tableau, des formes géométriques représentant des zones de couleurs de l'image recherchée. De cette façon, le système peut comparer la évaluer l'organisation (i.e. la disposition des couleurs) de l'œuvre d'art recherchée.
A titre d'exemple, la figure VI.6 montre l'outil de recherche par plan de QBIC. Cet outil permet à l'utilisateur de dessiner des zones de couleurs, à l'aide de formes rectangulaires ou elliptiques et une palette de couleur. De cette façon, le système pourra comparer la disposition des zones de couleurs avec l'organisation des œuvres d'art contenues dans la base de données. Les résultats renvoyés à l'utilisateur sont représentés de la même manière que lors d'une recherche par couleur. L'utilisateur a donc la possibilité d'effectuer les mêmes fonctions (affichage d'une œuvre d'art et recherche par similarité).

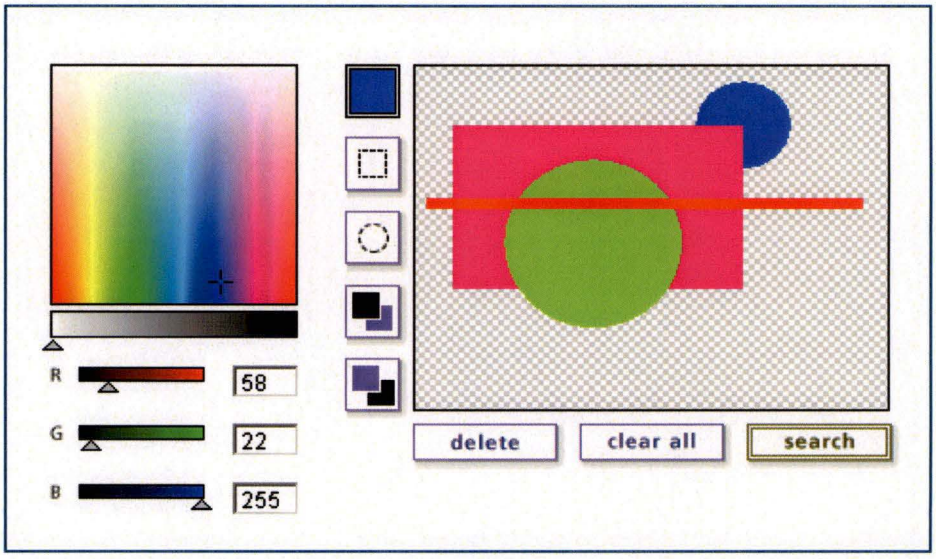


Figure VI.6. L'outil de recherche par plan de QBIC.

QBIC offre également une autre fonctionnalité qui n'est pas reprise sur le site du Musée de l'Hermitage: la recherche par la texture. Cette fonctionnalité permet de rechercher les images qui ont une texture semblable à une texture précisée par l'utilisateur.

Nous l'avons vu, une photographie peut être caractérisée par les personnes représentées sur l'image. Dès lors, on imagine aisément un système de recherche basé sur la reconnaissance de personnes, dans les images. C'est précisément ce que nous allons aborder dans la section suivante.

VI.2 – La reconnaissance des visages

De plus en plus, on observe l'importance de pouvoir retrouver de manière automatique la présence d'une personne particulière dans une base de données d'images ou de vidéos. Il y a quelque peu, la RTBF a réalisé un documentaire sur la relation qu'il

pouvait y avoir entre les différents membres du parti de l'extrême droite et l'appartenance à un groupe nazi. A cette fin, elle s'appuyait sur des anciennes vidéos sur lesquelles on pouvait voir certains membres du parti participer à des manifestations nazies antérieures.
Pour retrouver les vidéos en question dans l'immense banque de données de la télévision, une façon de faire est de consulter des fiches décrivant les séquences pour y trouver le nom des membres du parti. Toutefois, il se peut qu'à l'époque des manifestations nazies, ces personnes ne fussent pas encore connues. Dans ce cas, il n'y a aucune raison pour que leur nom apparaisse sur les fiches. Donc, une autre façon de faire serait d'employer une personne (ou plusieurs) qui consulterait une à une toutes les séquences vidéos concernant les manifestations nazies. Son but serait alors de retrouver parmi les images de la séquence le visage de l'un ou l'autre membre du parti, même en arrière plan. On imagine aisément que ce travail demande beaucoup de ressources en temps et en personnes. C'est pourquoi une solution dans laquelle les visages seraient reconnus de manière automatique serait plus intéressante pour de telles tâches. C'est à ce moment qu'intervient la recherche par la reconnaissance de visages.

La reconnaissance automatique de visages dans une image fixe ou une vidéo est une opération complexe dont on perçoit l'utilité dans le domaine de la prévention criminelle (vidéosurveillance, accès sécurisés par authentification, recherche d'identité d'un criminel sur base d'un portrait robot, etc.). En effet, le visage humain est une caractéristique biométrique, ce qui signifie que l'on peut reconnaître l'identité d'une personne grâce aux particularités de son visage (i.e. des caractéristiques géométriques ou statistiques dérivés de l'image du visage). C'est pourquoi la reconnaissance de visage est un domaine de recherche très actif. En règle générale, toutes les techniques de reconnaissance de visages valables pour la prévention criminelle le sont également pour l'identification de personnes dans des documents multimédia. Toutefois, il faut garder à l'esprit que, contrairement aux cas de la prévention criminelle, dans le cas d'un document multimédia:

- Le visage à identifier se trouvera rarement à la même position sur l'image.
- Le visage à identifier aura rarement la même taille.
- Le visage à identifier sera rarement présenté de la même façon sur l'image (il sera une fois vu de face, une autre fois de profil; ou une fois vu du haut et une autre fois vu du bas, etc.).
- On ne pourra jamais être certain de la présence (ou de l'absence) d'un visage dans une image.

Dès lors, dans le cas du multimédia, on rencontre une difficulté supplémentaire qui est d'identifier, dans les images, des zones susceptibles de représenter un visage, sans tenir compte de sa position, de sa taille ou de la façon dont il est présenté.

Selon [LIWE03], un système de reconnaissance de visages automatique comporte plusieurs tâches de traitement des visages telles que la détection de visages (*face detection*), le pistage de visages dans une séquence vidéo (*face tracking*), et l'évaluation de la similarité entre deux visages (i.e. la vérification et la reconnaissance de visages).

VI.2.1 – La détection de visages

La détection de visages dans une image consiste à identifier une forme (ou une zone), dans une image comme étant un visage potentiel. Il s'agit de la première étape d'un système de reconnaissance de visages automatique puisque, pour être reconnu, un visage doit d'abord être localisé.

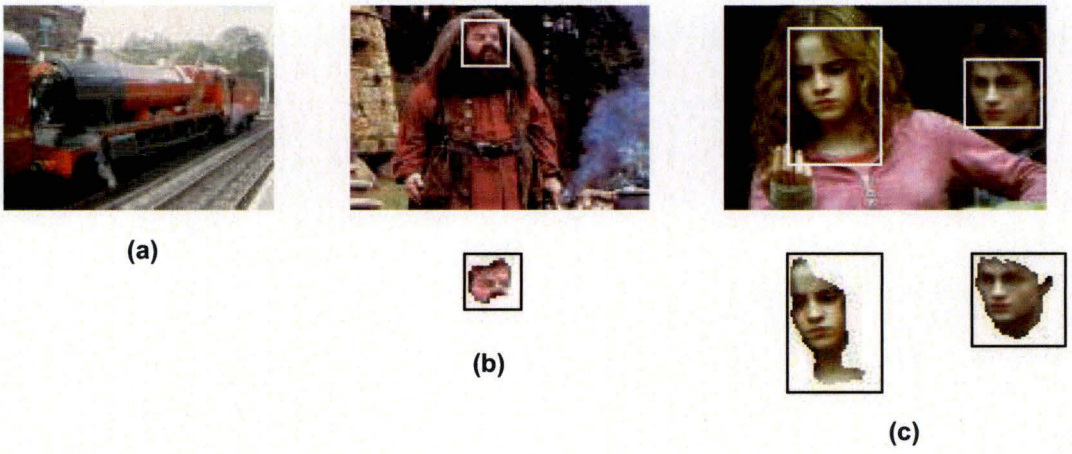


Figure VI.7. La détection de visages. **(a)** Une image ne contenant pas de visage. **(b)** et **(c)** deux images contenant des visages (en haut) et le résultat de la détection de ces visages: les zones de peau des visages (en bas).

La détection d'un visage dans une image fixe ne se fait pas exactement de la même façon que dans une vidéo. Dans le premier cas, la détection de visage doit se faire sur chaque image du système, pour être sûr de fournir à l'utilisateur tous les documents contenant la personne désirée. En revanche, dans le cas d'une vidéo, il n'est pas nécessaire d'effectuer une détection de visage pour chaque image constituant la vidéo. En effet, il y a de fortes chances que la personne que l'on recherche soit filmée pendant au moins une seconde. Dès lors, on peut être certain que le visage de la personne recherchée se retrouve au moins sur 24 images consécutives[6]. De plus, pour être pertinent, il faudrait que le résultat de la recherche permette à l'utilisateur de voir la personne désirée pendant une période de temps raisonnable. On pourrait donc définir un intervalle d'images dans lequel la détection de visage n'est pas appliquée. La longueur de cet intervalle doit permettre de détecter le plus de visages possible en assurant un temps de traitement raisonnable. Cela signifie que l'intervalle ne doit pas être trop grand ni trop petit. A titre d'exemple, le système Name-It[7] développé par [SNK99] applique la détection de visages toutes les 10 images.

L'approche la plus répandue pour la détection de visages dans une image utilise un réseau de neurones pour décider si un pixel fait partie d'un visage ou non. Toutefois, bien d'autres approches ont été abordées dans diverses recherches: détection du contour, détection des frontières et d'éléments particuliers, recherche de régions de couleurs, analyse de la texture, etc. Toutes ces approches correspondent, en réalité, aux différentes techniques d'analyse d'image dont [BJL03] consacre un chapitre de son ouvrage.
Comme nous pouvons le voir à la figure VI.7, le résultat de la détection d'un visage est une zone rectangulaire où l'on retrouve la majorité des pixels de la peau du visage, mais où les cheveux et l'arrière-plan sont effacés. Il est évident que, pour que la comparaison de visages puisse se faire, il est important de posséder des images de visages vus de face. C'est pourquoi les seules images retenues sont celles où l'on peut détecter les deux yeux. Pour ce faire, il faut que le détecteur de visage puisse également détecter les yeux.

[6] Rappelons que la norme du cinéma permettant de donner une animation fluide pour l'œil humain est de 24 images par seconde.
[7] Name-It est un système qui permet d'associer des visages et des noms dans les vidéos de journaux télévisés.

VI.2.2 – Le pistage de visages

Le pistage de visages ne s'applique qu'à la reconnaissance de visages dans des vidéos. Chaque visage détecté est pisté en avant et en arrière dans la vidéo, afin d'isoler la séquence d'images dans laquelle il apparaît. Ainsi, si l'on détecte un visage V à la 55ème image de la vidéo, on analysera les images 54 et précédentes jusqu'à ce que l'on arrive à une image (55-x) ne contenant plus le visage. De même, on analysera les images 56 et suivantes jusqu'à ce que l'on obtienne une image (55+y) sur laquelle on ne trouve plus le visage. De cette façon, on isole une séquence d'images [55-x, 55+y] (appelée "séquence de visages") contenant le visage V. Cette technique peut également être utilisée dans la vidéosurveillance, par exemple, pour suivre les mouvements d'un individu particulier.
Le pistage d'un visage se base sur le modèle de couleur de peau du visage détecté.

Extraction du modèle de couleur de peau

En général, le modèle de couleur de peau utilisé est un modèle Gaussien dans l'espace de couleurs (r, g) définit par le processus de normalisation illustré par les expressions VI.1 et VI.2.

$$r = R/(R + G + B) \qquad \textbf{[VI.1]}$$

$$g = G/(R + G + B) \qquad \textbf{[VI.2]}$$

Toutefois, cet espace de couleurs n'est pas vraiment sensible à la luminance de la couleur de la peau. C'est pourquoi il semble plus judicieux d'utiliser un modèle Gaussien dans l'espace de couleurs RGB.
Supposons que l'on ait détecté un visage V (composé de N pixels) et que $I(x, y)$ soit les intensités de couleurs $[R\ G\ B]^t$ en (x, y). Dès lors, un modèle de couleur de peau consiste en la matrice de covariance C que l'on peut voir à l'expression VI.3, la moyenne M que l'on peut voir à l'expression VI.4 et une distance d (constante).

$$M = \frac{1}{N} \sum_{(x,y) \in V} I(x,y) \qquad \textbf{[VI.3]}$$

$$C = \frac{1}{N} \sum_{(x,y) \in V} (I(x,y) - M)(I(x,y) - M)^t \qquad \textbf{[VI.4]}$$

Pour chaque visage détecté dans la vidéo, on construit un modèle de couleur de peau. Celui-ci sera utilisé pour identifier les pixels de peau candidats dans les images ultérieures, sur base de la propriété illustrée à l'expression VI.5 (le pixel $I(x, y)$ est un pixel de peau candidat s'il respecte la propriété).

$$(I(x,y) - M)^t C^{-1} (I(x,y) - M) < d^2 \qquad \textbf{[VI.5]}$$

Tous les pixels de peau candidats d'un même visage sont rassemblés afin de former une image binaire, de laquelle on supprime le bruit pour obtenir des zones de peau candidates. De cette façon, on peut évaluer le recouvrement de chacune de ces régions avec les zones de visage de l'image précédente pour décider si l'une d'entre elles est la zone de visage suivante. Le pistage d'un visage est effectué jusqu'à ce que l'on ne trouve plus de zone de visage correspondant au modèle de peau. A ce moment, on a isolé une séquence d'images

que l'on peut étiqueter et indexer à l'aide de la zone de peau du visage détecté. De cette façon, lorsqu'il faudra récupérer les séquences vidéo dans lesquelles on peut apercevoir un visage V, il suffira de rechercher la zone de peau qui ressemble le plus à celle de V et à retourner les séquences vidéo associées.

Sélection de la meilleure vue frontale du visage

Pour faciliter la comparaison des visages lors de la recherche, on peut imaginer que, au lieu d'étiqueter la séquence avec la zone de peau de départ (i.e. celle extraite lors de la phase de détection de visages), on sélectionne la zone de peau permettant la meilleure comparaison. Cela correspondrait à la meilleure vue frontale du visage, c'est-à-dire à l'image où l'on voit le visage le plus de face possible. Celle-ci peut être identifiée à l'aide de la position des yeux que le détecteur de visages aura localisés (soient (x_g, y_g) pour l'œil gauche et (x_d, y_d) pour le droit) et du centre de gravité de la zone de peau du visage (soit (x_c, y_c)): la meilleure vue frontale d'un visage correspond à la plus petite différence entre x_c et $(x_g + x_d)/2$, et à la plus petite différence entre y_g et y_d.

VI.2.3 – L'évaluation de la similarité entre deux visages

Il est important de faire la distinction entre les tâches de vérification et de reconnaissance de visages. La première trouve son utilité dans le domaine de l'accès par authentification: une personne se présente devant un système en prétendant avoir telle identité et celle-ci est vérifiée en comparant ses caractéristiques avec celles de l'individu qu'il prétend être. La reconnaissance de visage, quant à elle, est utilisée dans le cadre de la recherche de criminels sur base du portrait robot: le système recherche l'individu présentant des caractéristiques similaires à celles du portrait, de façon à attribuer son identité à la personne représentée sur le dessin. Dans le cas de la gestion de documents multimédia, la vérification de visage trouve son utilité dans le pistage de visages dans une vidéo et la reconnaissance de visages est utile lors de la recherche d'images et de vidéo basée sur le visage.

La tâche permettant d'évaluer la similarité entre deux visages consiste, en réalité, à comparer deux images entre elles. La mesure de la similarité entre deux images se fait grâce à la comparaison de caractéristiques que l'on peut en extraire: la couleur, la texture, la forme, etc. [BJL03] consacre à nouveau tout un chapitre de son ouvrage à la comparaison d'image. A l'heure actuelle, une des techniques les plus populaires pour l'évaluation de similarité entre deux visages est la reconnaissance basée sur les eigenfaces.

La reconnaissance de visages basée sur les eigenfaces

Dans le domaine de reconnaissance de visages, on appelle eigenfaces les caractéristiques du visage telles que le nez, la bouche, les yeux, etc. A titre d'illustration, la figure VI.8 montre des eigenfaces extraits d'un même visage initial. Chaque eigenface représente certaines caractéristiques que le visage de départ peut (ou peut ne pas) contenir. Selon [PIS03], ces caractéristiques peuvent être extraites du visage à l'aide d'une analyse en composantes principales (ou ACP). L'avantage de l'ACP est qu'elle permet la reconstruction de l'image initiale sur base des eigenfaces, si l'on additionne toutes les eigenfaces dans une bonne proportion. Si la caractéristique est bien représentée dans le visage original, l'eigenface correspondant prendra une grande importance dans la somme. En revanche, si la caractéristique n'est pas bien représentée dans le visage original (ou qu'elle n'est pas représentée du tout), l'eigenface correspondant prendra une moins grande importance dans

la somme (ou ne sera pas repris dans la somme). A cette fin, il faut associer à chaque eigenface un poids en fonction du taux de présence de la caractéristique dans le visage original. Notons qu'il est également possible de reconstruire une approximation du visage initial en utilisant seulement les eigenfaces les plus importants lors de la somme. C'est particulièrement utile lorsqu'il faut économiser des ressources informatiques.

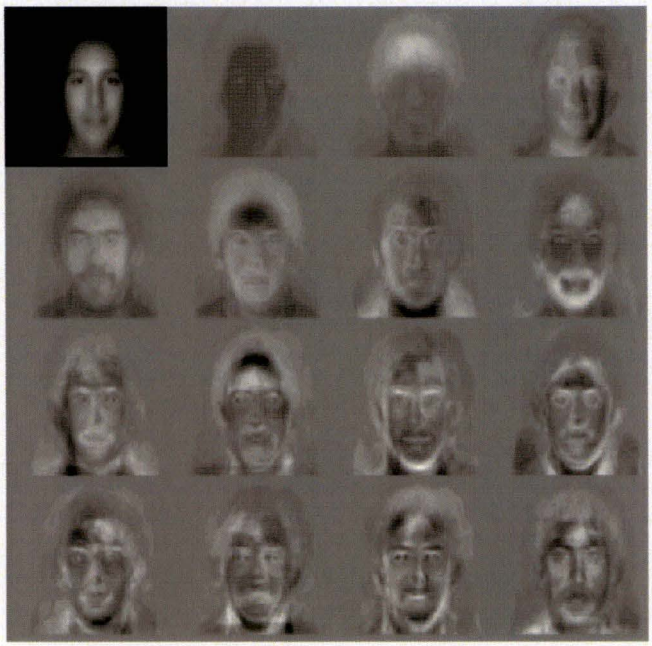


Figure VI.8. Les eigenfaces d'un même visage.

On imagine aisément comment les eigenfaces peuvent être utilisés pour la comparaison de visages: les visages similaires possédant des caractéristiques similaires, dans des proportions similaires (i.e. des eigenfaces similaires ayant des poids identiques), il suffit de comparer les poids des images (deux visages présentant des poids similaires peuvent être considérés comme semblables).

Conclusion

L'objectif de ce mémoire était de concevoir, par le biais d'un stage, un moteur de recherche permettant d'indexer des documents multimédia de manière efficace. Ce type d'outil vise la récupération et l'exploitation de documents parmi une grande quantité d'archives audiovisuelles. Il risque donc d'être d'une grande utilité pour des systèmes de gestion de documents multimédias tels que ceux accumulés par les musées ou les chaînes de télévision.

Le système de recherche décrit dans ce mémoire se base sur les descriptions structurées (i.e. des descripteurs XML) de documents multimédias. Cela nous a permis de faire un parallèle avec MPEG-7, le standard de description pour les contenus multimédia, basé sur les Schémas XML, qui nous est apparu intéressant pour ce type d'outil. Ce système s'articule donc autour de trois éléments principaux:

- Une base de données XML pour le stockage des descripteurs.
- Un outil permettant d'indexer et de récupérer les descripteurs (basé sur le moteur de recherche et d'indexation Lucene).
- Une banque de données de type WebDAV (i.e. HTTP étendu) pour le stockage des documents multimédia, permettant d'y accéder à distance.

Il permet ainsi de rechercher des documents audiovisuels, sur base de mots-clés susceptibles d'être présents dans leur description. Le grand inconvénient de ce système est que certaines choses ne peuvent pas être décrites avec des mots. Il nous a donc semblé intéressant de pouvoir offrir également à l'utilisateur le moyen d'approfondir ou d'effectuer sa recherche en indiquant (ou en fournissant) un document présentant des caractéristiques similaires à celui recherché. C'est pourquoi nous avons étudié les systèmes d'indexation et de recherche basées sur le contenu du document. Cela nous a permis de confirmer l'utilité de la norme MPEG-7 dans de tels systèmes. Nous avons également appris que, bien qu'une intervention humaine soit nécessaire à la description de la plupart des caractéristiques, d'autres peuvent être extraites de manière automatique, à l'aide d'algorithmes propres à chacune. Sachant cela, notre intérêt s'est tourné vers les systèmes de recherche d'images basée sur le contenu. Nous en avons donc étudié le fonctionnement avant de nous intéresser plus particulièrement à la fonction de recherche par reconnaissance de visages.
Un autre inconvénient du système, que nous avons pu mettre en évidence, vient de la complexité des outils qu'il met en œuvre. Par exemple, l'utilisation de Lucene a pour conséquence de consommer du temps puisqu'il faut convertir les descripteurs XML en input lors de l'indexation, et les résultats en output lors de la recherche. L'utilisateur ne serait-il pas autant satisfait des résultats fournis par le système, s'il avait à sa disposition des options de recherche plus simples telles que celles offertes par les bases de données XML? De même, le protocole d'échange utilisé permet d'éditer et de gérer collectivement des fichiers sur des serveurs Web distants. Les fonctions proposées par le protocole HTTP ne sont-elles pas suffisantes pour le type d'applications dans lequel s'inscrit le système? Il semblerait que la réponse à ces deux questions soit "oui".

Au terme de ce mémoire, il apparaît qu'un système efficace de recherche de documents multimédia doit être construit autour d'une base de données XML permettant de stocker et de gérer des descriptions MPEG-7. Le système disposerait donc d'un outil permettant de créer ces descriptions (off line) en extrayant des documents multimédia le plus de caractéristiques possible, de manière automatique (i.e. les caractéristiques de bas niveau). Les descriptions MPEG-7 pourraient alors être complétées manuellement, de façon à y introduire certaines caractéristiques ne pouvant pas être extraites automatiquement (i.e. les caractéristiques de haut niveau). Ce système doit permettre à l'utilisateur de retrouver le document désiré à l'aide de mots-clés correspondant typiquement aux caractéristiques de haut niveau. Pour des raisons évidentes de pertinence de résultats, ce système devrait aussi prendre en charge la correction orthographique ainsi que la synonymie des termes de la requête. L'utilisateur doit également avoir la possibilité d'interroger le système en décrivant

les caractéristiques de bas niveau du document recherché sur base de croquis (i.e. en dessinant ou en imitant les caractéristiques désirées) et/ou de document type (i.e. en indiquant ou en fournissant un document ayant des caractéristiques similaires à celles attendues). Dès lors le système utiliserait les caractéristiques du croquis ou du document type, afin de retrouver les descriptions contenant des caractéristiques semblables pour fournir les documents multimédia associés.
Il apparaît également qu'un tel système doit pouvoir être accessible via une interface Web permettant à l'utilisateur de formuler des requêtes, de visionner les résultats et de récupérer (télécharger) les documents qui l'intéressent.

Bien entendu, la conception d'un tel système nécessiterait une étude approfondie des caractéristiques auditives et visuelles pouvant être extraites automatiquement ainsi que de leurs techniques d'extraction. Il paraît évident que les caractéristiques nécessaires dans le système dépendent essentiellement des documents qu'il permet de gérer, des fonctions de recherche qu'il doit mettre en œuvre, et du type d'utilisateurs auquel il se destine. Quoi qu'il en soit, toutes ces caractéristiques pouvant être décrites à l'aide de descripteurs MPEG-7, nous espérons avoir réussi à montrer l'importance et l'utilité de cette norme pour les systèmes de gestion de documents multimédia.

SIGLES ET ABRÉVIATIONS

D (Descriptor) Descripteur MPEG-7. Il définit la syntaxe et la sémantique d'une caractéristique MPEG-7.

DCT (Discrete Cosine Transform) Transformée en cosinus directe. Elle permet de calculer la fréquence d'occurrence d'un pixel donné.

DDL (Description Definition Language) Langage de définition de description MPEG-7. Il permet notamment de créer de nouveaux Schémas de Description MPEG-7.

DOM (Document Object Model) Recommandation W3C décrivant un modèle de données modélisant les documents XML et une API permettant de manipuler ce modèle.

DS (Description Scheme) Schéma de Description MPEG-7. Il définit la syntaxe et la sémantique des relations entre les Descripteurs MPEG-7.

GED (Gestion Electronique de documents)

IEC (International Electronical Commission) Organisation internationale publiant des standards dans les domaines liés aux industries électriques. Intervient dans les domaines touchant à SGML et XML en association avec l'ISO.

ISO (International Organisation for Standardisation) Comité international de normalisation. Il regroupe environ 140 organismes nationaux de normalisation qui travaillent à des accords internationaux, appelés Normes.

JPEG (Joint Photographic Experts Group) Norme de compression pour les images fixes.

MPEG (Moving Picture Expert Group) Groupe de travail de l'organisme ISO/IEC. Ce comité a développé les standards MPEG-1, MPEG-2, MPEG-4, MPEG-7 et MPEG-21.

MPEG-1 Norme, développée par le groupe MPEG, pour le codage d'images animées et des sons associés pour le stockage de média jusqu'à 1,5 Mbit/s.

MPEG-2 Norme, développée par le groupe MPEG, pour le codage générique des images animées et des sons associés.

MPEG-4 Norme, développée par le groupe MPEG, pour le codage audio et vidéo en bas débit. L'usage principal est le Web, le CD, le vidéophone et la télévision.

MPEG-7 Norme, développée par le groupe MPEG, visant à décrire du contenu multimédia.

MPEG-21 Norme, développée par le groupe MPEG, destinée à tous les acteurs de la chaîne de diffusion et de consommation du document audiovisuel.

NXD (Native XML Database) Base de données XML native.

OCR (Optical Character Recognition) Système de reconnaissance d'écriture.

SAX (Simple API for XML) API "orientée flux" permettant à un parseur XML de communiquer aux applications les informations et la structure des documents XML au fur et à mesure de leur analyse syntaxique.

W3C (World Wide Web Consortium) Consortium publiant la plupart des spécifications relatives au World Wide Web.

XML (eXtensible Markup Language) XML est un méta-langage permettant de marquer la structure de documents texte de manière arborescente en insérant des "balises" dans le corps des documents.

XInclude (XML Inclusions) Mécanisme générique permettant de fusionner des documents XML dans un même et unique document XML.

XPath (XML Path Language) Langage de requête permettant d'accéder à chacun des éléments d'information d'un document XML, d'en sélectionner des listes et de les manipuler.

XPointer (XML Pointer) Langage permettant d'identifier des fragments XML dans des documents accessibles sur le Web.

XQuery (XML Query) Langage de requête permettant d'accéder à chacun des éléments d'information d'un document XML, d'en sélectionner des listes et de les manipuler. XQuery est un sur-ensemble de XPath.

XUpdate (XML Update) Langage de mise à jour de documents XML (suppression, insertion, mise à jour, interrogation, etc.) basé sur XPath et XSLT.

XSLT (XSL Transformations) Langage de programmation spécialisé permettant la transformation de documents XML. XSLT utilise XPath pour manipuler les éléments d'information de documents XML.

BIBLIOGRAPHIE

[BJL03] BRES S., JOLION J.-M. ET LEBOURGEOIS F., *Traitement et analyse des images numériques*, Lavoisier, Paris, France, 2003.

[BOU04] BOURRET R., XML and databases, http://www.rpbourret.com/xml/XMLAndDatabases.htm (dernière mise à jour en juillet 2004) (accédé le 31/07/2004).

[BRTI02] BRASSEUR C. et TIXHON R., *Analyse Technique des Formats MPEG*, mémoire présenté en vue de l'obtention du grade de Maître en Informatique, année académique 2001-2002.

[BUR+03] Burnet I., VAN DE WALLE R., HILL K., BORMANS J. et PEREIRA F., "MPEG-21: Goals and Achievements", *IEEE Multimedia*, pages 60-70, octobre-décembre 2003.

[CAL04] CALECA C., Le code ASCII, http://christian.caleca.free.fr/codage/le_code_ascii.htm (dernière mise à jour le 29/05/2004) (accédé le 10/07/2004).

[CAM99] CAMAPUM J.F., "Colour Recognition", in *Colour-based Recognition for Remote Sensing in Environmental Systems*, thèse de doctorat, pages 52-72, 1999.
Disponible à l'url http://www.ene.unb.br/~juliana/TesePhD/Chapter4.PDF
(accédé le 12/08/2004).

[CCM] COMMENT CA MARCHE (encyclopédie informatique libre), Le son numérique, http://www.commentcamarche.net/audio/son.php3
(accédé le 17/07/2004).

[CHA96] CHAUMIER J., *La gestion électronique de documents*, Presses Universitaires de France, Paris, France, 1996.

[CRDP04] CRDP DE GRENOBLE, L'image numérique, http://www.crdp.ac-grenoble.fr/image/general/img_num.htm (dernière mise à jour le 09/05/2004) (accédé le 10/07/04).

[DEL02] DELANNOY P., *WEBDAV en 2 minutes*, document daté de novembre 2002
Disponible à l'url http://www.univ-lemans.fr/~delannoy/DAV/webdavintro_fr.pdf
(accédé le 07/08/2004).

[DGLM03] DEMOULIN M., GOBERT D., LAZARO C. ET MONTERO E., "Que puis-je mettre sur mon site sans violer le droit des tiers", in *Guide pour les utilisateur d'Internet*, Hans D'HONT, Bruxelles, Belgique, 2003.

[DJE02] DJERABA C., "Content-Based Multimedia Indexing and Retrieval", *IEEE Multimedia*, pages 18-22, avril-juin 2002.

[EAGR99] EAKINS J.P. et GRAHAM M.E., *Content-based Image Retrieval: A report to the JISC Technology Applications Programme*, document daté de janvier 1999.
Disponible à l'url http://www.unn.ac.uk/iidr/CBIR/report.html
(accédé le 12/08/2004).

[EXIST] MEIER W., eXist: Open Source XML Database (page du projet eXist), http://exist-db.org (accédé le 02/08/2004).

[GAI02] GAIKWAD A., Introduction to Xindice, http://www-106.ibm.com/developerworks/web/library/wa-xindice.html (dernière mise à jour le 01/09/02) (accédé le 04/08/2004).

[GOR95] GORAY E., "Les technologies du multimédia", *Multimédia: Actes de la journée d'information sur le multimédia"*, Presses Universitaires de Namur, Namur, Belgique, 1995.

[HER94] HERELLIER J.-M., *Le Multimédia*, SYBEX, Paris, France, 1994.

[ISO01] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO), *MPEG-7 Requirements Document V.14*, document daté de mars 2001.
Disponible à l'url http://www.ipsi.fraunhofer.de/delite/projects/mpeg7/Documents/W4035.htm
(accédé le 9/07/04).

[ISO02a] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO), *MPEG-4 Overview – (V.21 – Jeju Version)*, document daté de mars 2002.
Disponible à l'url http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm
(accédé le 12/08/2004).

[ISO02b] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO), *MPEG-21 Overview v.5*, document daté d'octobre 2002.
Disponible à l'url http://www.chiariglione.org/mpeg/standards/mpeg-21/mpeg-21.htm
(accédé le 14/08/2004).

[ISO03] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO), *MPEG-7 overview (version 9)*, document daté de mars 2003.
Disponible à l'url http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm
(accédé le 9/07/04).

[JAKARTA] THE APACHE SOFTWARE FOUNDATION, The Jakarta Apache Project (page du projet Jakarta Lucene), http://jakarta.apache.org
(accédé le 05/08/2004).

[JPEG01] CHEZ.COM, Compression d'images fixes: norme JPEG,
http://www.chez.com/algorithmejpeg/dct.htm
(mis en ligne en 2001) (accédé le 19/07/2004).

[KEZU02] KEYVERS L. et ZUYLEN J.-B., *Création et exploitation d'archives audiovisuelles numériques*, mémoire présenté en vue de l'obtention du grade de Maître en Informatique, année académique 2001-2002.

[LIN03] LINWOOD J., Ajoutez un moteur de recherche à votre site grâce à Lucene,
http://www.zdnet.fr/builder/web_design/conception_site/0,39021086,39115855,00.htm
(mis en ligne le 19/08/2003) (accédé le 05/08/2004).

[LIWE03] LIU C. et WECHSLER H., "Independent Component Analysis of Gabor Features for Face Recongition", *IEEE Transactions Neural Networks*, vol. 14, no.4, pages 919-928, 2003.
Disponible à l'url http://www.cs.njit.edu/~liu/papers/tNN_1.pdf
(accédé le 23/08/2004).

[LOU00] LOUPIAS E., *Indexation d'images: aide au télé-enseignement et similarités pré-attentives*, thèse de doctorat, 2000.
Disponible à l'url http://telesun.insa-lyon.fr/~loupias/these/these_loupias.pdf
(accédé le 12/08/2004).

[MADIS] Page officielle du projet MADIS,
http://retina.crim.ca:8080/mpeg7-ric/home.do?action=index (accédé le 13/08/2004).

[MAR02a] MARTINEZ J., "MPEG-7, The Generic Content Description Standard, Part 1", *IEEE Multimedia*, pages 78-87, avril-juin 2002.

[MAR02a] MARTINEZ J., "MPEG-7, Overview of MPEG-7 Description Tools, Part 2", *IEEE Multimedia*, pages 83-93, juillet-septembre 2002.

[MEI02] MEIER W., "eXist: An Open Source Native XML Database", in *Akmal B. Chaudri, Mario Jeckle, Erhard Rahm, Rainer Unland (Eds.): Web, Web-Services, and Database Systems. NODe 2002 Web- and Database-Related Workshops*, Erfurt, Germany, October 2002. Springer LNCS Series, 2593.
Disponible à l'url http://exist-db.org/webdb.pdf
(accédé le 02/08/2004).

[MON04] MONTERO E., *Informatique et droit*, notes du cours INFO 2322: Droit de l'informatique, année académique 2003-2004.

[MPEG02] TECHNISCHE UNIVERSITÄT CHEMNITZ: FALKUTAT FÜR INFORMATIK, MPEG Video Compression Technique, http://rnvs.informatik.tu-chemnitz.de/~jan/MPEG/HTML/mpeg_tech.html
(dernière mise à jour le 21/08/02) (accédé le 19/07/2004).

[PEL99] PELLETIER F., Introduction à la GEIDE, http://www.mosarca.com/acro/ged99.pdf
(document daté de 1999) (accédé le 20/01/2004).

[PIS03] PISSARENKO D., Eigenface-based facial recognition,
http://openbio.sourceforge.net/resources/eigenfaces/eigenfaces-html/facesOptions.html
(document daté du 13/02/2003) (accédé le 25/08/2004)

[QBIC] QBIC™ – IBM's Query By Image Content, http://wwwqbic.almaden.ibm.com/
(accédé le 14/08/2004)

[SGU04] ŞAYKOL E., GÜDÜKBAY U. ET ULUSOY Ö., *Integrated Querying of Images by Color, Shape, and Texture Content of Salient Objects*, to appear in *Advances in Information Systems (ADVIS'04)*, Izmir, Turkey, October 2004.
Disponible à l'url http://www.cs.bilkent.edu.tr/~ediz/bilmdg/papers/advis04.pdf
(accédé le 13/08/2004).

[SNK99] SATOH S., NAKAMURA Y et KANADE T., "Name-It: Naming and Detecting Faces in News Videos", *IEEE Multimedia*, pages 22-35, janvier-mars 1999.

[STA01] STAKEN K., Introduction to Native XML Databases,
http://www.xml.com/pub/a/2001/10/31/nativexmeldb.htm
(mis en ligne le 31/10/2001) (accédé le 31/07/2004).

[TAN04] TAN S.S., Exposés Système sur le thème de l'opensource: Analyse de la structure de Lucene, http://www-igm.univ-mlv.fr/~dr/XPOSE2003/lucene/articleLucene.html
(dernière mise à jour le 28/02/2004) (accédé le 05/08/2004).

[TIS99] THE INTERNET SOCIETY, HTTP Extensions for Distributed Authoring – WEBDAV,
http://asg.web.cmu.edu/rfc/rfc2518.html
(document daté de février 1999) (accédé le 07/08/2004).
Traduction française: THOMASSON J.-J., Extensions de HTTP pour la rédaction distribuée,
http://www.webdav.org/specs/rfc2518.fr.html
(accédé le 07/08/2004).

[UCI99] UNIVERSITY OF CALIFORNIA, IRVINE (UCI), WebDAV,
http://www.univ-lemans.fr/~delannoy/DAV/webdav_flyer.pdf
(document daté de février 1999) (accédé le 07/08/2004).

[UCI01] UNIVERSITY OF CALIFORNIA, IRVINE (UCI), WEBDAV: Collaborative Document Authoring and Management, http://ftp.ics.uci.edu/pub/ietf/webdav/intro/webdav_flyer.pdf
(dernière mise à jour le 19/04/2001) (accédé le 08/08/2004).

[UNICODE] UNICODE INC, About the Unicode Standard, http://www.unicode.org/standard/standard.html
(accédé le 10/07/1004).

[UTZ04] UTZ WESTERMANN G., *PTDOM – a Persistent Typed Document Object Model for the Management of MPEG-7 Media Descriptions*, these de doctorat, 2004.

[VAR02] VARANDAT M., Stockage de contenus XML: base "native" ou relationnelle?, http://www.indexel.net/bin/doc/1997 (mis en ligne le 06/11/2002) (accédé le 30/07/2004).

[VETA00] VELTKAMP R.C. et TANASE M., *Content-Based Image Retrieval Systems: A Survey*, tecnhical report, octobre 2000. Disponible à l'url http://ftp.cs.uu.nl/pub/RUU/CS/techreps/CS-2000/2000-34.pdf (accédé le 12/08/2004).

[WEBDAV] STEIN G., Site de la communauté WebDAV, http://www.webdav.org/ (accédé le 08/08/2004).

[XINDICE] THE APACHE SOFTWARE FOUNDATION, Apache Xindice (page du projet Xindice), http://xml.apache.org/xindice/index.pdf (accédé le 03/08/2004).

[XPATH] WORLD WIDE WEB CONSORTIUM (W3C), *XML Path Language (XPath) 2.0*, Working Draft du W3C, juillet 2004. Disponible à l'url http://www.w3.org/TR/xpath20/ (accédé le 08/08/2004).

[XQUERY] WORLD WIDE WEB CONSORTIUM (W3C), XQuery 1.0: An XML Query Language, Working Draft du W3C, juillet 2004. Disponible à l'url http://www.w3.org/TR/xquery/ (accédé le 08/08/2004).

[XSLT] WORLD WIDE WEB CONSORTIUM (W3C), XSL Transformations (XSLT) Version 2.0, http://www.w3.org/TR/xslt20/ (accédé le 08/08/2004).

[XMLa] WORLD WIDE WEB CONSORTIUM (W3C), *XML Schema Part 1: Structures*, Recommendation W3C, mai 2001. Disponible à l'url http://www.w3.org/TR/xmlschema-1 (accédé le 10/07/2004).

[XMLb] WORLD WIDE WEB CONSORTIUM (W3C), *XML Schema Part 2: Datatypes*, Recommendation W3C, mai 2001. Disponible à l'url http://www.w3.org/TR/xmlschema-2/ (accédé le 10/07/2004).